

Revista mensual 1987 Precio 375 Ptas Año 2 Número 17

ENSAMBLADOR PARA TU SPECTRUM

TODO SOBRE LOS NUMEROS ROMANOS

OTELO, UN JUEGO DE INGENIO

COMO MANEJAR FICHEROS



AÑO 2 NUMERO 17

DIRECTOR: Manuel Pérez
COORDINADOR EDITORIAL: Francisco de Molina
DIRECTOR DE ARTE: Luis F. Balaguer
REALIZACION GRAFICA: Didac Tudela
COLABORADORES: José Vila, Ernesto del Valle, Equipo Molisoft, Ramón Rabaso, Antonio Taratiel, Jaime Mardones
FOTOGRAFIA: Ernesto Wallisch, Joan Boada

INPUT Sinclair es una publicación de PLANETA-DE AGOSTINI, S.A.
GERENTE DIVISION DE REVISTAS:
Sebastián Martínez

PUBLICIDAD: José Real-Grupo Jota
Madrid: c/ General Varela, 35
Teléf. 270 47 02/03
Barcelona: Avda. de Sarriá, 11-13, 1.°
Teléf. 250 23 99

FOTOMECANICA: TECFA, S.A.

IMPRESION: Sirven Grafic
C/ Gran Vía, 754-756, 08013 Barcelona
Depósito legal: B. 38.115-1986

SUSCRIPCIONES: EDISA
López de Hoyos, 141, 28002 Madrid
Teléf. (91) 415 97 12

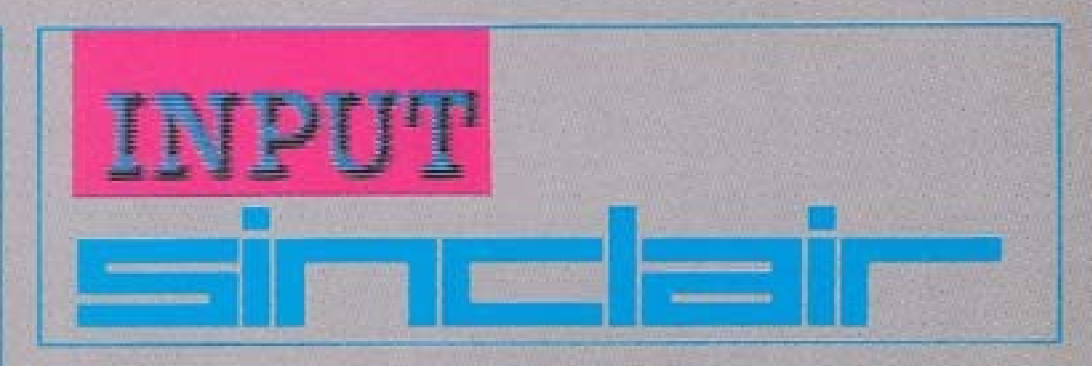
REDACCION:
Aribau, 185, 1.°
08021 Barcelona

DISTRIBUIDORA:
R.B.A. PROMOTORA DE EDICIONES, S.A.
Travesera de Gracia, 56. Edificio Odiseus.
08006 Barcelona

El precio será el mismo para Canarias que para la Península y en él irá incluida la sobretasa aérea.

INPUT Sinclair es una publicación controlada por





SUMARIO

UNA GRAN NOTICIA

Este mes ofrecemos una gran noticia a nuestros lectores. A iniciativa de una de las más importantes empresas españolas de software, desde el primer día de marzo los precios de los videojuegos comerciales van a bajar radicalmente su precio.

Por las informaciones de que disponemos en el momento de cerrar este número, la iniciativa ha encontrado una acogida positiva entre el resto de empresas del sector, muchas de las cuales han anunciado su intención de adherirse.

Es obvio que para nuestros lectores ésta no es una novedad cualquiera, pues se trata de un descenso de precios que, en algunos casos, puede situarlos en tan sólo una cuarta parte del que antes tenían en el mercado.

Sin embargo, las consecuencias de esta medida rebasan este aspecto, aun siendo fundamental. Además de acabar con un mecanismo que mantenía los precios artificialmente altos y, por tanto, perjudicaba de forma injustificada al usuario, con las nuevas tarifas el mercado tenderá a ser más dinámico, más receptivo a nuevas ideas e iniciativas y, como consecuencia, a expandirse. Hay que esperar que esta mayor demanda redunde beneficiosamente sobre la producción nacional de software y la industria de comercialización que se genera en torno a ella.

Las nuevas tarifas entran en vigor a partir del mismo día en que esta revista se pone a la venta. Estamos ante una gran noticia para los jóvenes aficionados a los videojuegos y que, en muchos casos, no disponían del poder adquisitivo necesario.

Sin duda, entre usuarios, productores y distribuidores existe conciencia de que la producción interna de software necesitaba de este tipo de medidas para desarrollarse y abandonar su estado de dependencia.

Esperemos que así sea.

LOS MEJORES DE INPUT

Hemos pensado que es interesante disponer de un ***ranking*** que ponga en claro, mes a mes, cuáles son los programas preferidos de nuestros lectores. Para ello, es obligado preguntaros directamente y tener así el mejor termómetro para conocer vuestras preferencias. Podéis votar por cualquier programa aunque no haya sido comentado todavía en **INPUT**.

El resultado de las votaciones será publicado en cada número de **INPUT**.

Entre los votantes sortearemos 10 cintas de los títulos que pidáis en vuestros cupones.

Nota: No es preciso que cortéis la revista, una copia hecha a máquina o una simple fotocopia sirven.

Enviad vuestros votos a: **LOS MEJORES DE INPUT** Aribau, 185. Planta 1. 08021 Barcelona

ELIGE TUS PROGRAMAS

Primer título elegido ____________________

Segundo título elegido ____________________

Tercer título elegido ____________________

Programa que te gustaría conseguir ____________________

Qué ordenador tienes ____________________

Nombre ____________________

1.er Apellido ____________________

2.º Apellido ____________________

Fecha de nacimiento ____________________

Teléfono ____________________

Dirección ____________________

Localidad ____________________

Provincia ____________________

INPUT SINCLAIR N.º 17

MUSICA Y SIMULACION DE PARTITURAS

A continuación publicamos la segunda parte del listado correspondiente al artículo *Música y simulación de partituras.*

```
1Ø5Ø IF a$="4" THEN GO TO 16ØØ
11ØØ GO TO 9Ø5
1199 REM subrutina de play
12ØØ GO SUB 5ØØ
121Ø FOR n=1 TO max
122Ø PRINT PAPER 5;AT M(n,3),12;CHR$ (M(n,4))
1225 IF M(n,5)=2 THEN PRINT AT M(n,3),11;"."
123Ø IF M(n,5)=1 THEN PRINT AT M(n,3),11;"b"
1235 IF M(n,2)=1ØØ THEN PAUSE M(n,1)*t*50: GO TO 125Ø
124Ø BEEP M(n,1)*t,M(n,2)
125Ø PRINT PAPER 5;AT M(n,3),11;" "
127Ø NEXT n
1275 LET n=max: LET cv=5: GO SUB 5ØØ
128Ø RETURN
1299 REM edit
13ØØ LET a$="EDIT MODE s SALIR x ADELANTE z ATRASmMODIFICAR n NEW": GO SUB 5ØØØ:
     IF mod=1 THEN GO TO 9Ø5
131Ø PRINT AT M(n,3),12; PAPER 5; FLASH 1; CHR$ (M(n,4)): IF M(n,5)=1 THEN PRINT PAPER
     5;AT M(n,3),11;"b"
1312 IF M(n,5)=2 THEN PRINT PAPER 5;AT M(n,3),11;"."
1315 PAUSE Ø: PRINT AT M(n,3),12; PAPER 5, FLASH Ø; OVER 1;" "
132Ø LET a$=INKEY$
1325 PRINT AT M(n,3),11; PAPER 5;" "
133Ø IF a$="x" AND n<max THEN LET n=n+1
135Ø IF a$="z" AND n>1 THEN LET n=n-1
136Ø IF a$="s" THEN LET editmode=Ø: LET mod=Ø; LET cv=5: LET ctrlcp=gctr: GO SUB 5ØØ:
     GO TO 9ØØ
137Ø IF a$="m" THEN PRINT AT M(n,3),11; PAPER 5;" ": LET gmax=max: LET ctrlcp=Ø: LET
     editmode=1: LET n=n-1: LET mod=1: LET fv=M(n,3): LET cv=12: GO TO 9Ø5
1375 IF a$="n" THEN RUN
138Ø GO TO 131Ø
1399 REM print
14ØØ LET mod=Ø: LET cv=5: LET n=max: LET editmode=Ø: GO SUB 5ØØ: LET ctrlcp=gctr: LET
     a$="AHORA ESTAS EN EL CONTROL DE TECLADO": GO SUB 5ØØØ: RETURN
1499 REM save
15ØØ LET a$="PREPARA LA CASSETTE PARA GRABAR": GO SUB 5ØØØ
151Ø DIM V(21): LET V(2Ø)=am: LET V(21)=ctrlcp: LET V(1)=max: LET V(2)= compas: LET
     V(3)=gc: LET V(4)=octava: LET V(5)=t: LET V(6)=(t-1)/.Ø9+193: FOR a=1 TO 13: LET
     V(6+a)= CODE SCREEN$ (a,3): NEXT a
152Ø SAVE "musica" DATA M(): SAVE "armadura" DATA Z(): SAVE "adicional" DATA V()
1525 CLS: GO SUB 4ØØ
153Ø RETURN
```

```
1599 REM load
1600 LET a$="PREPARA LA CASSETE PARA CARGAR": GO SUB 5000
1610 LOAD "musica" DATA M(): LOAD "armadura" DATA F(): LOAD "adicional" DATA V()
1615 LET am=V(20): LET ctrlcp=V(21): LET max=V(1): LET compas=V(2): LET gc=V(3): LET
     t=V(5): LET octava=V(4): LET tabsc=V(6)
1620 CLS: GO SUB 400
1625 FOR a=1 TO 13: PRINT AT a,3; PAPER 5; CHR$ (V(6+a)): NEXT a
1630 GO SUB 4510
1640 GO SUB 1830: GO TO 850
1699 REM octava
1700 IF a$<>"H" THEN GO TO 1740
1710 FOR b=octava TO 2
1720 FOR a=1 TO max: IF M(a,2)<>100 THEN LET M(a,2)=M(a,2)+12
1730 NEXT a: NEXT b
1735 LET octava=3: GO TO 1830
1740 IF a$<>"L" THEN GO TO 1780
1750 FOR b=octava TO 2 STEP -1
1760 FOR a=1 TO max: IF M(a,2)<>100 THEN LET M(a,2)=M(a,2)-12
1770 NEXT a: NEXT b: LET octava=1: GO TO 1830
1780 IF octava=2 THEN GO TO 1830
1790 IF SGN (octava-2)=1 THEN LET suma=-12
1800 IF SGN (octava-2)=-1 THEN LET suma=12
1810 FOR a=1 TO max: IF M(a,2)<>100 THEN LET M(a,2)=M(a,2)+suma
1820 NEXT a: NEXT b: LET octava=2
1830 PRINT AT 9,22; PAPER 6; BRIGHT 0; OVER 1;" ";AT 9,21+octava; BRIGHT 1;" "
1840 RETURN
1899 REM tiempo
1900 IF a$="o" AND tabsc-1<>182 THEN OVER 1: GO SUB 4600: OVER 0: LET
     tabsc=tabsc-1: LET t=t-.09: GO SUB 4600: PRINT AT 5,26; PAPER 6; INVERSE 1;" ";AT
     5,26; tabsc-183
1910 IF A$="p" AND tabsc+1<>248 THEN OVER 1: GO SUB 4600: OVER 0: LET tabsc=tabsc+1:
     LET t=t+.09: GO SUB 4600: PRINT AT 5,26; PAPER 6; INVERSE 1;" ";AT 5,26;
     tabsc-183
1920 RETURN
1999 REM control de pentagrama
2000 PRINT PAPER 7;AT 9,28;" "
2005 IF a$="w" AND fv-1<>0 THEN PRINT PAPER 5;AT fv,cv; FLASH 0; OVER 1;" ";AT fv-1,
     cv; FLASH 1;" ": LET fv=fv-1: LET u=n2+1: IF u=8 THEN LET u=1
2010 IF a$="s" AND fv+1<>14 THEN PRINT PAPER 5;AT fv,cv; OVER 1; FLASH 0;" ";AT fv+1,
     cv; FLASH 1;" ": LET fv=fv+1: LET u=n2-1: IF u=0 THEN LET u=7
2015 PRINT PAPER 7;AT 9,28;N$(u)
2020 RETURN
2099 REM obtencion de ff a partir de fv
2100 LET n1=14-fv
2110 IF n1>=8 THEN LET n2=n1-7: LET ff=F(n2)+12
2120 IF n1<8 THEN LET n2=n1: LET ff=F(n2)
2130 RETURN
2499 REM codificacion de beeps
2500 LET n=n+1: LET max=n: LET am=am+1: IF am=15 THEN GO SUB 4800
2510 IF B(n2)=1 THEN LET M(n,5)=1
2520 IF B(n2)=2 THEN LET M(n,5)=1: LET M(n,3)=fv-1: GO TO 2540
```

```
2530 LET M(n,3)=fv
2540 LET rt=1/(2↑(rcd−97))
2550 IF gcd=1Ø3 THEN LET rt=rt*1.5: IF M(n,5)<>1 THEN LET M(n,5)=2: LET gcd=Ø
2560 LET M(n,1)=rt
2570 IF sl=1 THEN LET ff=1ØØ
2580 LET M(n,2)=ff
2590 IF cd>=97 AND cd<=103 THEN LET M(n,4)=CODE z$(cd−96)
2600 IF cd>=65 AND cd<=71 THEN LET M(n,4)=CODE z$((cd−64)+5)
2610 RETURN
3999 REM printer
4ØØØ PRINT PAPER 5;AT fv,cv;CHR$ (M(n,4))
4Ø2Ø LET cv=cv+2
4Ø3Ø IF cv=21 THEN LET cv=5: GO SUB 55Ø
4Ø4Ø LET ctrlcp=ctrlcp+M(n,1): IF ctrlcp>=compas THEN GO SUB 42ØØ
4Ø5Ø RETURN
4199 REM compas lleno
42ØØ LET ctrlcp=Ø: FOR a=1 TO 13: PRINT AT a,cv−1; PAPER 5; OVER 1;"|"
421Ø NEXT a
422Ø FOR a=1 TO 7: LET B(a)=Ø: NEXT a
423Ø GO SUB 451Ø
424Ø RETURN
4499 REM pasos Z(a)−F(a)
45ØØ FOR a=1 TO 7: LET Z(a)=F(a): NEXT a: RETURN
451Ø FOR a=1 TO 7: LET F(a)=Z(a): NEXT a: RETURN
4599 REM puntero de tiempo
46ØØ PLOT tabsc−2,142: DRAW 4,Ø: DRAW −2,2: DRAW −1,−1: DRAW 1,Ø: RETURN
4799 REM ampliacion de matriz
48ØØ LET am=Ø: DIM H(lm,5): FOR a=1 TO lm: LET H(a,1)=M(a,1): LET H(a,2)=M(a,2): LET
     H(a,3)=M(a,3): LET H(a,4)=M(a,4): LET H(a,5)=M(a,5): NEXT a
481Ø LET lm=lm+15: DIM M(lm,5)
482Ø FOR a=1 TO lm−15: LET M(a,1)=H(a,1): LET M(a,2)=H(a,2): LET M(a,3)=H(a,3): LET M(a,
     4)=H(a,4): LET M(a,5)=H(a,5): NEXT a
483Ø RETURN
4999 REM subrutina de mensajes
5ØØØ FOR a=1 TO 2Ø: FOR b=1 TO 3: PRINT AT 14+b,a; PAPER Ø;" ": NEXT b: NEXT a
5ØØ5 LET m$="                              ": LET m$(1 TO LEN a$)=a$
5Ø1Ø PRINT PAPER 8; INK 7;AT 15,1;m$(1 TO 2Ø);AT 16,1;m$(21 TO 4Ø);AT 17,1;m$(41 TØ
     6Ø)
5Ø2Ø RETURN
9999 PAPER 7: BORDER 7: CLS
```

APRENDE A DIBUJAR EN 3-D (y II)

El programa de dibujo de modelos de alambre que hemos visto en los dos artículos anteriores te permite dibujar un cubo en tres dimensiones y cambiar su tamaño. Pero te presenta siempre el mismo punto de vista; la cara frontal del cubo es la que aparece siempre más próxima a ti y, aunque puedes ver a través del sólido de alambre, no tienes posibilidad de verlo desde arriba, desde un lado o desde donde quieras mirarlo. En este artículo encontrarás unas rutinas adicionales que te permitirán especificar la posición de tu ojo de manera que puedas contemplar el cubo desde cualquier dirección.

Se trata de una herramienta muy útil, especialmente cuando empieces a dibujar objetos más complicados, ya que al contemplarlos según diferentes direcciones puedes descubrir rasgos ocultos o semiescondidos. Además, si especificas una sucesión de coordenadas diferentes para la posición de tu ojo, puedes obtener la impresión de estar *sobrevolando* el objeto o caminando alrededor del mismo. Sin embargo, la secuencia de dibujos que es posible obtener en un ordenador doméstico es muy lenta y el efecto no es el mismo que se obtiene con la velocidad de los sólidos de alambre comerciales en que el observador parece que se precipita sobre el coche, planeta o lo que sea, a gran velocidad. No obstante, los principios que intervienen son básicamente los mismos.

EL EFECTO DEL PUNTO DE VISTA

Ya vimos en el artículo anterior cómo es posible dibujar un objeto *tridimensional* en una pantalla de dos dimensiones utilizando una convención visual que puede ser interpretada por el ojo y el cerebro del observador. Los programas anteriores hacían esto utilizando una proyección *isométrica* en la que las líneas inclinadas se supone que retroceden o avanzan a partir de la pantalla.

La forma más común de convención visual es el dibujo en perspectiva, en el que las líneas que se alejan del observador convergen en un punto distante, llamado punto de fuga, acortándose en escorzo como si *desaparecieran*. En una verdadera perspectiva de tres puntos hay realmente tres direcciones de fuga y tres puntos de fuga, una para cada uno de los tres ejes. La posición de los tres puntos está determinada por la relación entre la posición del observador y la posición del objeto que se desea representar. (Recuerda sin embargo que en la convención visual estos puntos son imaginarios.)

Con esto ya disponemos de una clave sobre la forma de representar un objeto, tal como un cubo en un espacio de tres dimensiones, tal como se vería desde diferentes puntos de vista. Para ello será necesario definir unas nuevas transformaciones que produzcan el efecto de convergencia hacia un punto de fuga y que acorten el objeto.

Lo primero que tienes que saber es la relación que hay entre la posición del observador y la del objeto. Para determinarla tienes que relacionar el punto de vista con los ejes X, Y y Z que has definido en tu pantalla. Eso es precisamente lo que hacen los programas que vamos a ver, ocupándose después de las transformaciones necesarias.

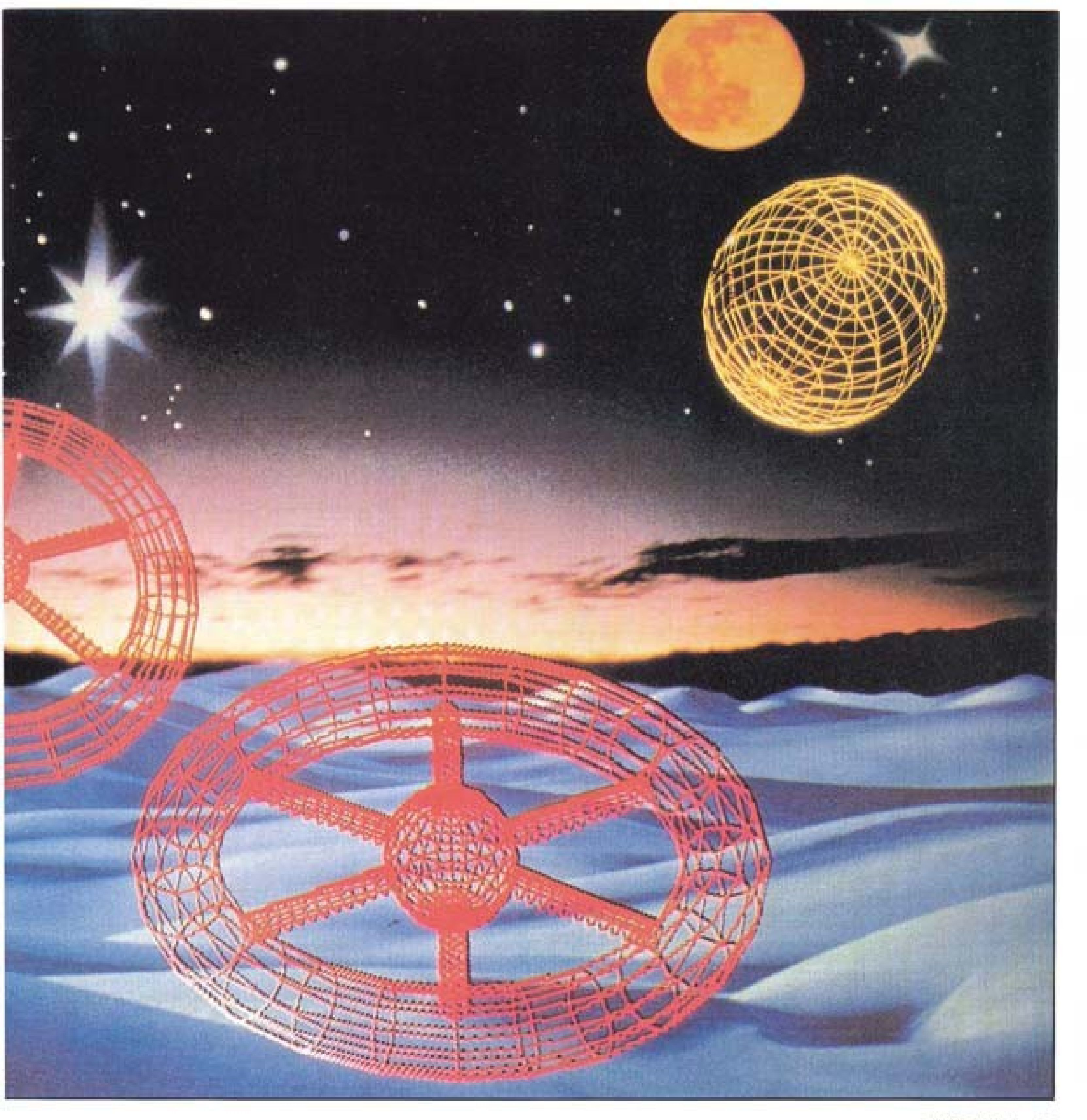
ANTES DE EMPEZAR

Para poder ejecutar las nuevas secciones del programa, necesitas la *rutina de rejilla* que vimos en el artículo anterior. Si guardaste una copia de dicha rutina, tendrás ahora que cargarla con una instrucción LOAD. Sería una



¡Impresionante! Ésta es la única manera de calificar la potencia transformadora de este programa, el segundo de la serie, en el que se incorporan la perspectiva, el punto de vista y la generación de figuras alámbricas.

- EL EFECTO DEL PUNTO DE VISTA
- ESTABLECIENDO LAS VARIABLES INICIALES
- PROGRAMACIÓN MODULAR



buena idea que cargaras también la *rutina de dibujo del círculo* que también explicamos. En realidad ahora no la vas a necesitar, pero conviene tener todas las rutinas juntas para el programa de dibujo del globo que veremos más adelante. En consecuencia las líneas que necesitas son desde la 5000 a la 6160; puedes borrar todas las demás líneas.

Igual que antes, tendrás que teclear varias secciones de programa antes de poder pasar a ejecutar nada, ya que primeramente tienes que definir la posición del ojo y las transformaciones para las rotaciones y la perspectiva.

DEFINIENDO TU PUNTO DE VISTA

La primera sección del programa calcula las variables necesarias para determinar la posición de tu ojo en el espacio tridimensional:

```
8000 LET XV=X: LET YV=Y: LET
     ZV=Z
8010 LET WV=YV*YV+ZV*ZV
8020 LET PV=SQR (XV*XV+WV)
8030 IF PV=0 THEN RETURN
8035 LET WV=SQR WV
8040 LET XU=XV/PV
8050 LET YU=YV/PV
8060 LET ZU=ZV/PV
8070 LET WU=WV/PV
8080 REM ORIENTACION
8090 LET A=XV*YV: LET B=ZV:
     GO SUB 8450: LET G=H
8100>LET A=YV:LET B=XV:GO
    SUB 8450:LET G=G+H
8110 LET SG=SIN G
8120 LET CG=COS G
8140 LET R1=WU*CG
8150 LET R2=-WU*SG
8160 LET R3=-XU
8170 LET R4=-YU
8180 LET R5=-ZU
8190 LET R6=XV*XU+YV*YU+
     ZV*ZU
8200 IF WU=0 THEN GO TO
     8340
8210 LET XT=XV*WU-(YV*YU+
     ZV*ZU)*XU/WU
8220 LET YT=(YV*ZU-ZV*YU)/
     WU
8230 LET R7=(ZU*SG-
     XU*YU*CG)/WU
8240 LET R8=(-YU*SG-
     XU*ZU*CG)/WU
8250 LET R9=CG*XT+SG*YT
8260 LET S1=(ZU*CG+
     XU*YU*SG)/WU
8270 LET S2=(-YU*CG+
     XU*ZU*SG)/WU
8280 LET S3=-SG*XT+CG*YT
8330 RETURN
```



```
8350 LET R7=-1
8360 LET R8=0
8370 LET R9=0
8380 LET S1=0
8390 LET S2=1
8400 LET S3=0
8410 RETURN
8450 IF B<>0 THEN LET
     H=ATN (A/B): RETURN
8460 LET H=PI/2: RETURN
```

Para entender lo que va sucediendo, recuerda que el eje Z avanza hacia ti saliendo desde el centro de la pantalla, el eje Y apunta hacia arriba de la pantalla y el eje X apunta hacia la derecha de la pantalla.

El ojo se supone situado en (XV, YV, ZV), donde la V significa punto de Vista. La variable WV da la distancia en la dirección resultante de combinar Y y Z. Se supone que el objeto que estás contemplando está situado en el origen de coordenadas (0,0,0), origen que por simplicidad suponemos coincidente con el centro de la pantalla. La línea 8030 fuerza la salida de la rutina por la vía rápida en el caso en que el punto de vista, es decir la posición del ojo, sea el origen, ya que es difícil que mires a tu propio ojo, como no sea con un espejo. Las variables XU, YU y ZU son las distancias en las direcciones de los tres ejes X, Y, Z de una línea de longitud unidad dibujada entre el origen y la posición del ojo. WU es la distancia resultante en la dirección combinada de X y Z.





Cuando contemplas un objeto, puedes obtener diferentes vistas del mismo, moviéndote a su alrededor de izquierda a derecha. Para ello se mide un ángulo a partir del eje X, tal como se hace en las líneas 8090 y 8100. Hemos incluido una comprobación (líneas 8450 y 8460) para evitar la división por cero, que interrumpiría el programa y te enviaría un mensaje de error.

Con esto se obtiene una vista normal cuando se mira directamente a lo largo del eje X, produciéndose un giro gradual a medida que te vas moviendo alrededor, con lo que resultan vistas más interesantes. Las líneas 8140 a 8280 definen las variables que fijan la orientación del ojo en el espacio. La posición del ojo se fija de manera que su eje Z permanece sobre la línea que une el ojo con el origen de la pantalla.

El resto de la rutina define las variables para el caso especial en que la posición del ojo está realmente sobre el eje X.

LAS TRANSFORMACIONES

La siguiente sección transforma las coordenadas X, Y, Z del cubo en coordenadas finales sobre la pantalla. Estas transformaciones tienen en cuenta la posición del ojo y el efecto de perspectiva:

```
8500 LET X1=T1*X+T4*Y+T7
8510 LET Y1=T2*X+T5*Y+T8
8520 LET Z1=T3*X+T6*Y+T9
8540 LET X2=R1*X1+R7*Y1+
     R8*Z1+R9
8550 LET Y2=R2*X1+S1*Y1+
     S2*Z1+S3
8560 LET Z2=R3*X1+R4*Y1+
     R5*Z1+R6
8575 IF Z2<ZN THEN RETURN
8580 LET X3=D*X2/Z2
8590 LET Y3=-D*Y2/Z2
8600 RETURN
```

Esta rutina utiliza el álgebra matricial, algo complicada, para transformar las coordenadas (X,Y), aunque básicamente se trata de los pasos de transformación descritos en el primer artículo de esta serie. Las líneas 8500 a 8520 transforman el plano (X,Y) de dibujo en dos dimensiones en el espacio (X1, Y1, Z1) de tres dimensiones. Las líneas 8540 a 8560 transforman las coordenadas (X1, Y1, Z1) del espacio tridimensional posicionándolas con arreglo a la situación y dirección del ojo (X2, Y2, Z2). La línea 8575 comprueba si la nueva posición que hay que dibujar está demasiado cerca de la posición del ojo, es decir, si Z2 se encuentra comprendido entre 0 y ZN, o si está detrás del ojo (Z2 negativo). En ambos casos la posición de dibujo se ignora, ya que sería imposible visualizar el objeto. Si la posición está más alejada que ZN, contada a partir de la posición del ojo, al final de la rutina se calculan las coordenadas de la posición sobre la pantalla (X3, Y3). El parámetro D/Z2 que figura en estas líneas incorpora la perspectiva a la imagen, al proyectar el objeto sobre una pantalla plana a una distancia D del ojo. Si asignas a D un valor pequeño, resultará el efecto de mover la pantalla cerca del ojo, con lo que el efecto de perspectiva es más acusado. Cuando D es grande, la pantalla está lejos y el efecto de perspectiva es pequeño, la imagen aparece virtualmente sin distorsión, incluso aún cuando se contempla oblicuamente en una dirección cualquiera.

ASIGNACIÓN DE LAS VARIABLES INICIALES

El siguiente paso es escribir de nuevo las *rutinas de Dibujo* e *Inicialización* para poder hacer uso de las nuevas *rutinas de Transformación*:

```
9000 CLS
9020 LET XM=256: LET
     YM=176
9030 LET XD=XM/2: LET
     YD=YM/2
9040 LET ZN=1
9042 INPUT "INTRODUCE LA
     DISTANCIA AL PLANO DE
     PROYECCION",D
9045 IF D<=0 THEN LET
     D=1000*ZN
9050 LET T1=1: LET T2=0 LET
     T3=0
9060 LET T4=0: LET T5=1: LET
     T6=0
9070 LET T7=0: LET T8=0: LET
     T9=1
9090 CLS : RETURN
9500 LET X=XS: LET Y=YS: GO
     SUB 8500: IF Z2<ZN
     THEN GO TO 9520
9505 IF X3<-127 OR Y3<-87
     OR X3>128 OR Y3>88
     THEN GO TO 9550
9510 PLOT 127+X3,87+Y3
9520 LET X=XE: LET Y=YE: GO
     SUB 8500: IF Z2<ZN
     THEN GO TO 9550
9525 IF X3<-127 OR Y3<-87
     OR X3>128 OR Y3>88
```

```
      THEN GO TO 955Ø
953Ø  DRAW 127+X3-PEEK
      23677,87+Y3-PEEK
      23678
955Ø  RETURN
```

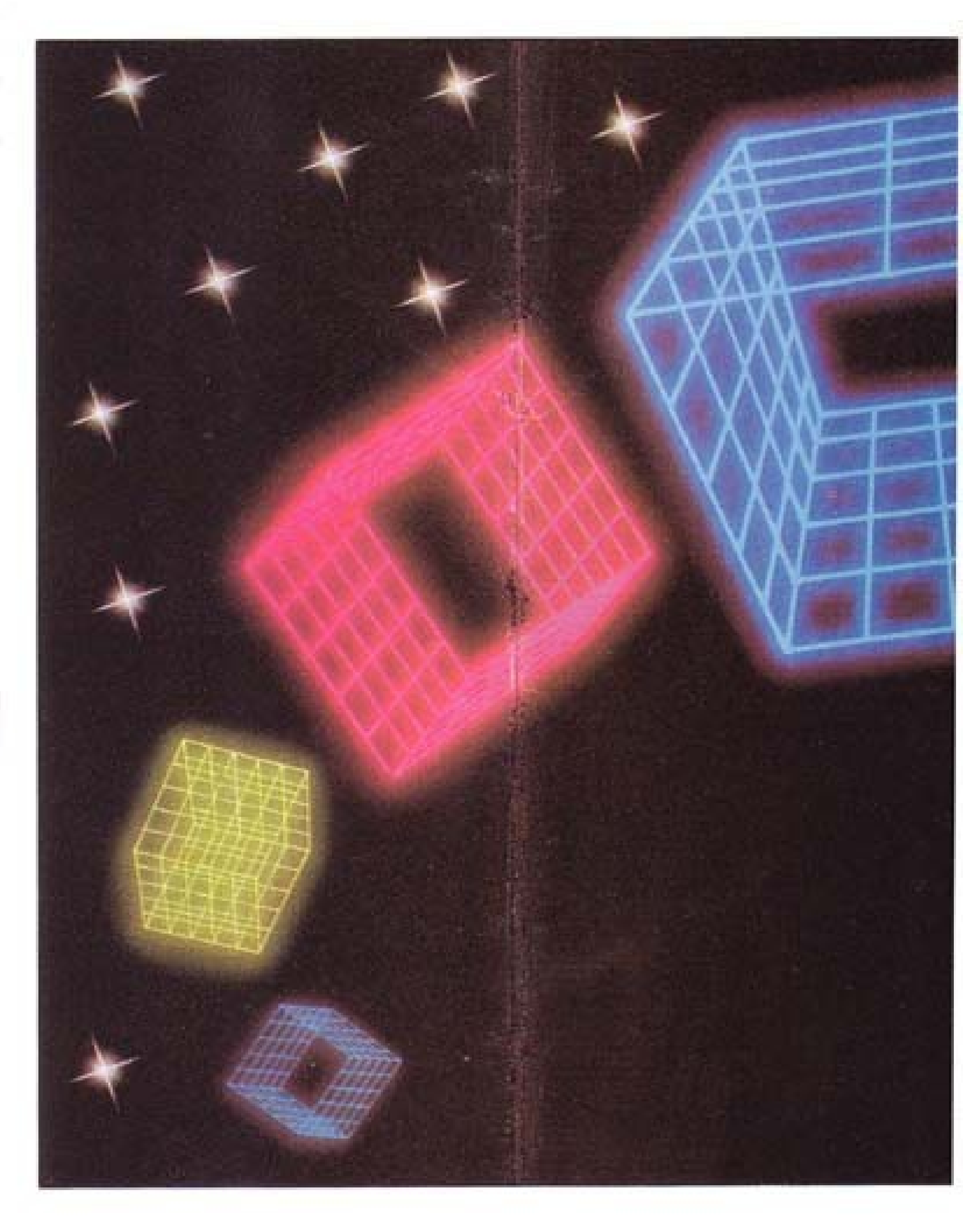
La línea 9020 define las dimensiones máximas de la pantalla en las direcciones X e Y, mientras que la línea 9030 define el punto medio. La línea 9040 fija la posición más próxima al ojo, que se permite a los puntos del dibujo. La variable D da la distancia real desde la posición del ojo hasta el plano de proyección, es decir hasta la pantalla, con lo que determina la perspectiva. Los valores de D se asignan durante la fase de ejecución del programa, con lo cual puedes modificar el grado de perspectiva. Si no se introduce ningún valor, al pulsar ENTER RETURN la línea 9045 establece un valor por defecto de 1000. Las líneas 9050 a 9070 se ocupan de pasar los valores a las constantes de transformación, para especificar en el espacio de tres dimensiones el plano en que hay que dibujar la imagen.

Esta sección del programa continúa con la versión revisada de la rutina de dibujo. En primer lugar se definen las constantes de la transformación (línea 9500) y a continuación se hace una comprobación (líneas 9505 y 9525) para determinar si hay que dibujar un nuevo punto en la pantalla (líneas 9510 y 9530). Esta comprobación se hace necesaria para evitar el mensaje de error que resultaría si intentas pintar la pantalla.

LLAMANDO A LA RUTINA

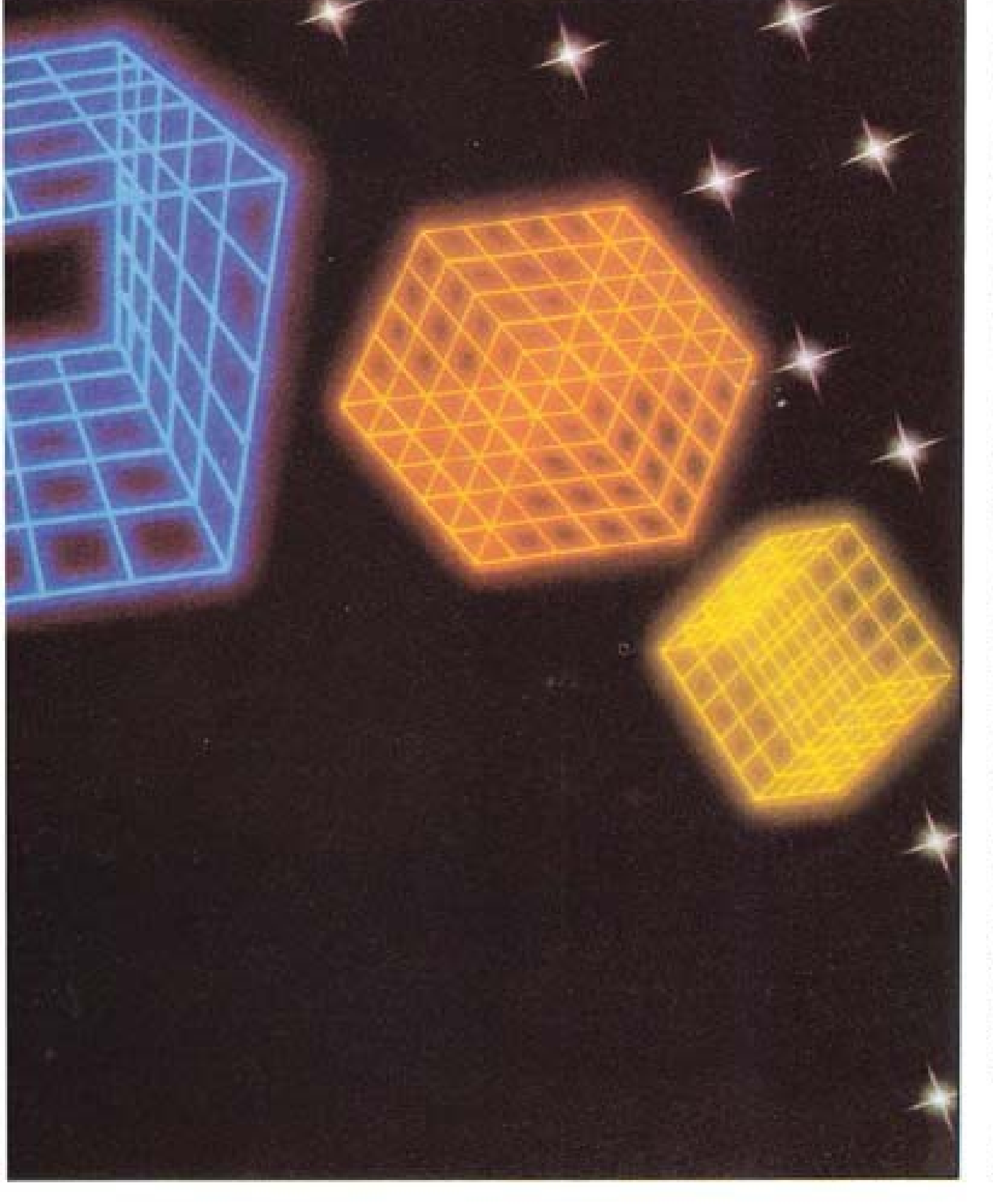
Ahora necesitas la rutina para dibujar la retícula. Si no tecleaste el programa del artículo anterior, hazlo ahora, junto con este programa.

```
11Ø  GO SUB 9ØØØ
12Ø  LET L=2Ø: LET N=3
125  GO SUB 5Ø5: GO TO 14Ø
13Ø  GO SUB 5ØØ
14Ø  IF X=Ø AND Y=Ø AND Z=Ø
     THEN GO TO 17Ø
15Ø  GO SUB 1ØØØ
16Ø  GO TO 13Ø
17Ø  CLS
18Ø  STOP
5ØØ  IF INKEY$="" THEN GO TO
     5ØØ
51Ø  INPUT "INTRODUCE LA
     POSICION DEL OJO (X,Y,Z)",
     X,Y,Z
52Ø  GO SUB 8ØØØ
53Ø  RETURN
1ØØØ  LET P=L/2
1Ø1Ø  LET T1=1: LET T2=Ø: LET
      T3=Ø
1Ø2Ø  LET T4=Ø: LET T5=1: LET
      T6=Ø
1Ø3Ø  LET T7=-P: LET T8=-P:
      LET T9=-P
1Ø4Ø  GO SUB 12ØØ: REM
      ABAJO
1Ø5Ø  LET T7=-P: LET T8=-P:
      LET T9=P
1Ø6Ø  GO SUB 12ØØ: REM
      ARRIBA
1Ø7Ø  LET T4=Ø: LET T5=Ø: LET
      T6=-1
1Ø8Ø  GO SUB 12ØØ: REM
      IZQUIERDA
1Ø9Ø  LET T7=-P: LET T8=P:
      LET T9=P
11ØØ  GO SUB 12ØØ: REM
      DERECHA
111Ø  LET T1=Ø: LET T2=-1:
      LET T3=Ø
112Ø  GO SUB 12ØØ: REM
```



```
    ATRAS
1130 LET T7=P: LET T8=P: LET
    T9=P
1140 GO SUB 1200: REM
    ADELANTE
1170 RETURN
1200 LET XA=0: LET YA=0: LET
    LW=L: LET LH=L: LET
    NX=N: LET NY=N
1210 GO SUB 5000
1220 RETURN
```

Puedes ejecutar ahora el programa. Si funciona correctamente, entrará a la *rutina de Inicialización* que empieza en la línea 110. Esta rutina define algunas variables y presenta un mensaje de pantalla pidiéndote que introduzcas un valor para D, la distancia al plano de proyección. La perspectiva incorporada en el programa es tal que cuanto mayor es esta distancia, más lejos aparece el objeto con lo que resultará menos efecto de perspectiva. Para empezar puedes introducir un valor de 1000. El programa retorna a la línea 120, que especifica la longitud L de cada lado del cubo, y el número N de cuadrados de la rejilla en cada cara. Las líneas 130 a 160 leen las coordenadas de la posición del ojo (línea 510), dibujando a continuación el cubo a partir de esa posición. En cuanto esté dibujada la primera vista, puedes introducir un nuevo conjunto de coordenadas, para ver el cubo desde una nueva dirección. El programa termina cuando se introducen los valores 0, 0 y 0. La rutina que abarca las líneas 500 a 530 hace realmente una llamada a la *rutina de Posición* (línea 520) para establecer las constantes de la transformación. La *rutina del Cubo* (líneas 1000 a 1170) es la que posiciona a continuación y dibuja cada una de las seis caras. La *rutina Cara* situada entre las líneas 1200 y 1220 dibuja cada cara como una cuadrícula, con ayuda de la *rutina Grid*.



Ensaya diferentes valores de la posición del ojo para estudiar el efecto del punto de vista. Para empezar, puedes utilizar un valor de 1000 para D y 200, 0 y 0 para X, Y, Z. El cubo aparece con el mismo tamaño pero la perspectiva es mucho más pronunciada. Para introducir nuevos valores de X, Y y Z, pulsa cualquier tecla y aparecerá el mensaje de pantalla. Para modificar el valor de D, tienes que pulsar BREAK y a continuación ejecutar el programa de nuevo. Te aparecerá un mensaje pidiéndote el valor de D y un segundo mensaje pidiéndote los valores de X, Y y Z.

Si en algún momento hubiera que dibujar puntos fuera de la zona de pantalla, te aparecerá un mensaje de error. En consecuencia el programa ignora cualquier línea que se salga fuera de la zona disponible. Esto puede llevar a resultados extraños, ya que se omite toda una línea incluso aunque sólo sea una pequeña parte de ella la que se salga del máximo permitido. Los puntos que se encuentren más próximos que una determinada distancia mínima, y los puntos que queden por detrás del ojo, se ignoran. Así, si la posición del ojo es muy cercana al cubo o se encuentra dentro del mismo, pueden producirse también resultados falsos. Se podría incluir una comprobación de estos puntos, haciendo un recorte aproximado de las líneas, pero esto requeriría una considerable cantidad de programa y de cálculo adicional.

Guarda ahora en cinta o en disco una copia del listado completo, ya que a continuación verás la forma de usar

estas mismas rutinas para dibujar formas duplicadas y para generar algunos gráficos circulares bastante espectaculares.

MODELOS DE ALAMBRE Y CURVAS

Hasta ahora hemos estado viendo la forma de conseguir figuras básicas a partir de retículas rectangulares y círculos concéntricos. También hemos visto cómo se pueden utilizar diferentes transformaciones para combinar una serie de rejillas que componen la imagen de un cubo en perspectiva. En este artículo introducimos en el programa que hemos desarrollado hasta ahora unas variaciones mínimas que nos permitirán ampliar la gama de posibles figuras de alambre que se pueden dibujar.

Para empezar, es muy sencillo conseguir imágenes múltiples. Si has trabajado con el programa que empieza en la línea 8000, habrás observado que se obtienen efectos de perspectiva bastante espectaculares cuando D (la distancia al observador) adquiere valores pequeños, por ejemplo entre 100 y 400, mientras que el efecto de perspectiva es muy poco acusado cuando D es del orden de varios miles. Puedes continuar ahora con el siguiente ejercicio para visualizar no uno, sino cuatro cubos al mismo tiempo en la pantalla.

OBJETOS MÚLTIPLES

Si conservas aún el programa del artículo anterior, te ahorrarás un montón de trabajo utilizando una simple instrucción de LOAD a fin de cargarlo de nuevo en tu ordenador. Si no lo guardaste, tendrás que teclearlo de nuevo. Asegúrate también de que tienes la *rutina Circulo*, que vimos en el primer artículo.

Como todo el programa está escrito en bloques o rutinas autónomos, resulta sencillo modificarlo para que dibuje figuras múltiples en vez de dibujar una sola. Por ejemplo, para dibujar cuatro cubos, no tienes más que especificar su posición en la pantalla y llamar a la *rutina Cubo* cuatro veces. El mejor sitio para especificar la posición es la rutina de transformación de coordenadas. Modifica esta parte del listado de la manera siguiente, pero no ejecutes el programa todavía:

```
8500 LET X1=T1*X+T4*Y+
     T7+XO
8510 LET Y1=T2*X+T5*Y+
     T8+YO
8520 LET Z1=T3*X+T6*Y+
     T9+ZO
```

Las variables XO, YO y ZO al final de estas líneas especifican un desplazamiento en cada una de las tres direcciones de los ejes de coordenadas a fin de determinar la separación de los cuatro cubos. Estas variables son las que reciben valores justo antes de que se dibuje cada cubo.

Teclea ahora la siguiente sección del programa y ejecútalo para observar el efecto del desplazamiento:

```
 120 LET L=20: LET PP=20:
     LET N=1
 150 GO SUB 1500
1500 LET XO=-PP: LET
     YO=-PP: LET ZO=0
1520 GO SUB 1000
1530 LET XO=PP: LET
     YO=-PP: LET ZO=0
1540 GO SUB 1000
1550 LET XO=PP: LET YO=PP:
     LET ZO=0
1560 GO SUB 1000
1570 LET XO=-PP: LET
     YO=PP: LET ZO=0
1580 GO SUB 1000
1590 RETURN
```

Al ejecutar el programa, tienes que ir respondiendo a los mensajes que te aparecen en la pantalla. El primer mensaje te pide que introduzcas un valor para D, la distancia al plano de proyección. Un valor de 1000 es bueno para empezar. El siguiente mensaje te pide que introduzcas la posición del ojo, por lo que tendrás que teclear los valores de X, Y y Z. Aquí te presentamos un conjunto de valores que te darán determinadas imágenes en la pantalla. Prueba con ellos y observa cómo se distorsionan las imágenes cuando el valor de D es pequeño, y cómo con D grande la distorsión es inapreciable. Ensaya otros valores a tu gusto, incluyendo alguno de varios miles para D, así como valores negativos para X, Y y Z.

D	X	Y	Z
100	55	0	0
900	200	-250	200
20000	1000	3000	10000

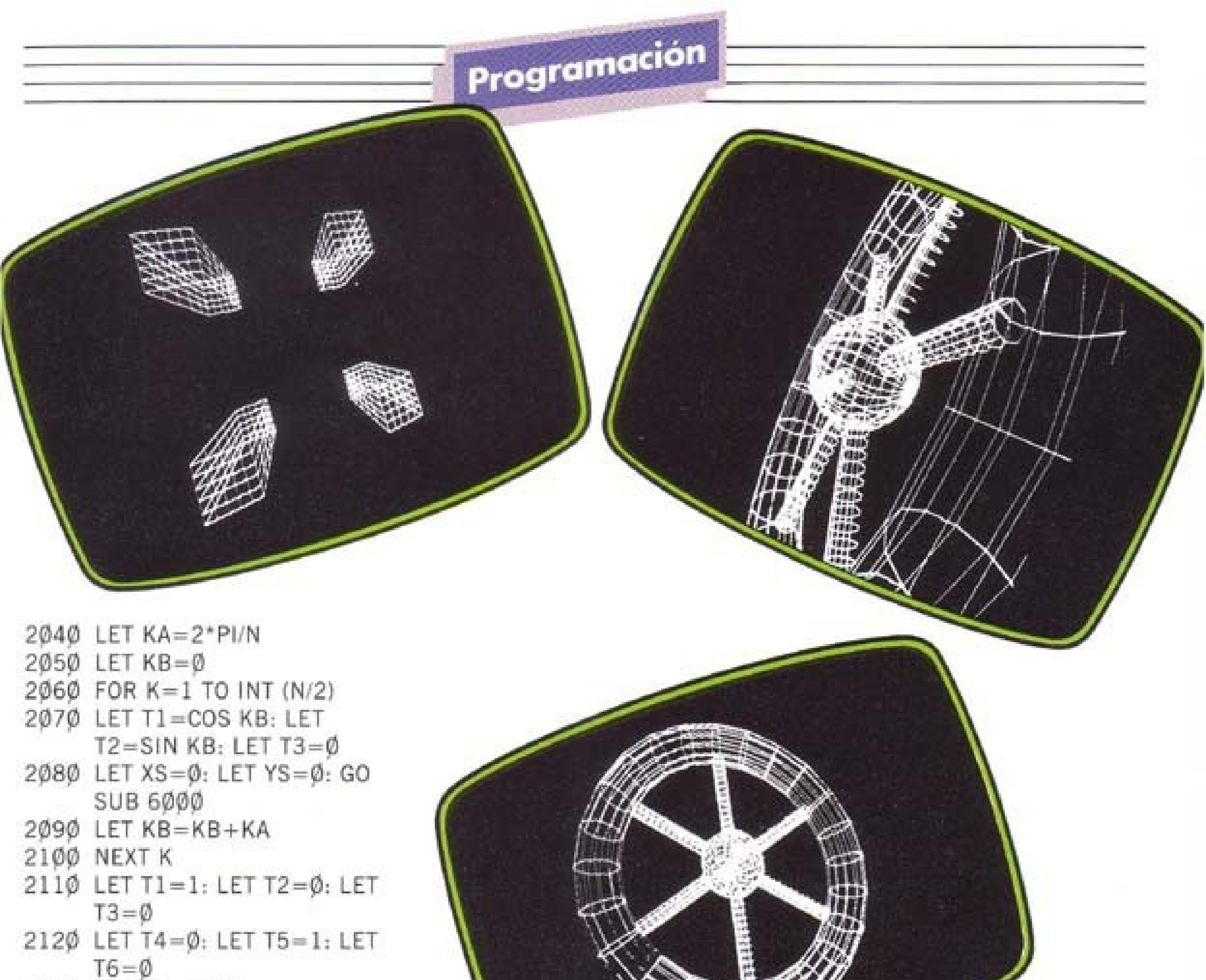
La rica variedad de vistas posibles con este programa te abre ya amplios horizontes para realizar experimentos interesantes, pero no tienes por qué ceñirte únicamente a cambiar los valores de la posición del ojo. Como habrás visto, la línea 120 de la última sección de programa que has tecleado define como variables la longitud de la arista del cubo (L), la separación de un cubo a otro (PP) y el número de cuadrados de la retícula (N) para cada una de las caras; aquí tienes pues otras tres variables que puedes modificar antes de hacer correr el programa.

DIBUJANDO UN GLOBO

Aunque el programa que acabas de introducir es corto, es capaz de configurar imágenes complejas, debido a que puede llamar a cualquiera de las rutinas que componen el resto del listado. Una de estas rutinas podría ser la que dibuja círculos concéntricos, y que ya comentamos en el artículo anterior. Si tu programa no contiene esta rutina, tecléala ahora y guarda una copia del programa completo para poder utilizarla en el futuro.

Ahora puedes modificar el programa para que en vez de llamar a las *rutinas del cubo, retícula y cara*, llame a la *rutina Círculo*. Borra la línea 120 y a continuación teclea las siguientes líneas:

```
 100 LET XO=0: LET YO=0:
     LET ZO=0
 150 LET R=20: LET N=18: GO
     SUB 2000
2000 LET T7=0: LET T8=0: LET
     T9=0
2010 LET T4=0: LET T5=0: LET
     T6=1
```



```
2Ø4Ø  LET KA=2*PI/N
2Ø5Ø  LET KB=Ø
2Ø6Ø  FOR K=1 TO INT (N/2)
2Ø7Ø  LET T1=COS KB: LET
      T2=SIN KB: LET T3=Ø
2Ø8Ø  LET XS=Ø: LET YS=Ø: GO
      SUB 6ØØØ
2Ø9Ø  LET KB=KB+KA
21ØØ  NEXT K
211Ø  LET T1=1: LET T2=Ø: LET
      T3=Ø
212Ø  LET T4=Ø: LET T5=1: LET
      T6=Ø
213Ø  LET KA=KA/2
214Ø  LET KB=Ø: LET RR=R
215Ø  FOR K=1 TO N
216Ø  LET T7=Ø: LET T8=Ø: LET
      T9=RR*COS KB
217Ø  LET XS=Ø: LET YS=Ø: LET
      R=RR*SIN KB: GO SUB
      6ØØØ
218Ø  LET KB=KB+KA
219Ø  NEXT K
22ØØ  RETURN
```

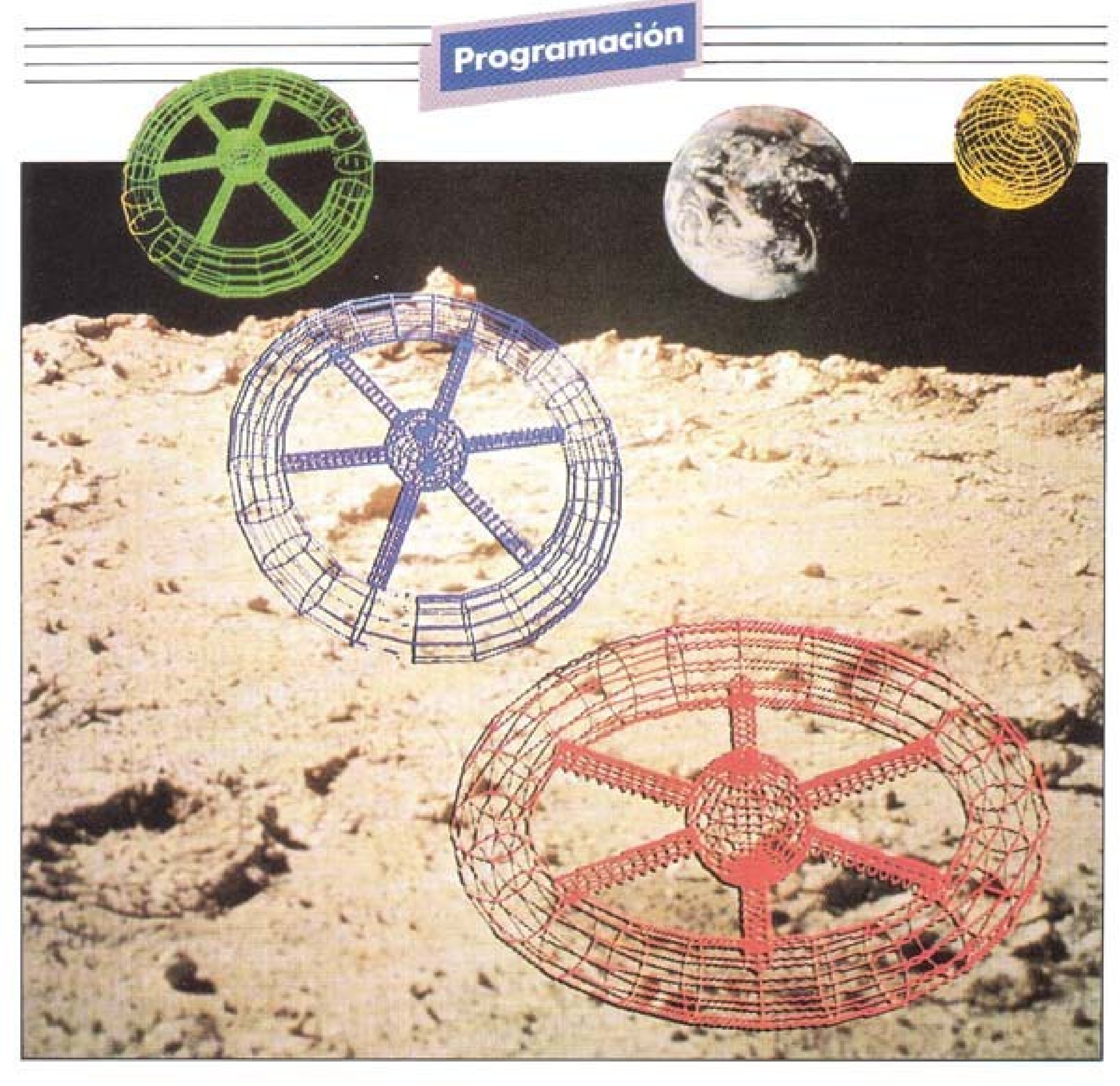
La primera parte de la fase de ejecución de este programa te resultará familiar, ya que se trata de seleccionar un valor de D y a continuación un valor para la posición del ojo. Una vez introducidos estos valores, la línea 150 llama a la rutina recién tecleada para dibujar la imagen de un globo. Esto se hace a partir de una serie de elipses (o círculos, dependiendo de la posición del ojo). Por una parte están las líneas de longitud (que pasan por los polos de la esfera) y por otra las de latitud, que son paralelas al ecuador.

El valor del radio de la esfera está dado por la variable R (línea 150), mientras que el número de líneas de longitud y latitud está dado por N, que tiene que ser un número par mayor que 2. Las líneas 2000 a 2050 definen las variables para posicionar el globo y se utilizan además como contadores. El bucle que hay entre las líneas 2060 y 2100 se ocupa de dibujar las líneas de longitud (los meridianos), mientras que el que va desde la línea 2150 a la 2190, dibuja las líneas de latitud constante (los paralelos).

El eje de la esfera es la línea que pasa por los polos y su dirección depende de los valores que hayas introducido. Para ver la forma en que quedan situados los globos en relación con los ejes de la pantalla, teclea el valor 0 para la posición de dos de las coordenadas del punto de vista, por ejemplo (0,0,200) y observa el resultado. Prueba ahora a introducir valores positivos o negativos para las tres coordenadas, para ver cómo se desplazan los polos respecto a los ejes.

PASANDO DESDE EL GLOBO AL TORO

El globo es una de las formas más sencillas que puede representarse me-



diante una serie de elipses, pero este mismo programa cuenta con todos los elementos necesarios para dibujar otras formas mucho más exóticas como por ejemplo un toro (es decir, un *donut* completo con un agujero en medio). Para ello no se requieren más que unas pequeñas modificaciones en la rutina del globo. Aquí las tienes:

```
 15Ø LET R=1Ø: LET N=18:
     LET RT=2Ø: LET N2=18:
     GO SUB 2ØØØ
2Ø4Ø LET KA=2*PI/N2: LET
     NC=N2
2Ø45 IF RT=Ø THEN LET
     NC=INT (NC/2)
2Ø6Ø FOR K=1 TO NC
2Ø8Ø>LET XS=RT:LET YS=Ø:GO
     SUB 6ØØØ
213Ø LET KA=2*PI/N
2135 IF RT=Ø THEN LET
     KA=KA/2
217Ø LET XS=Ø: LET YS=Ø: LET
     R=RT+RR*SIN KB: LET
     N=N2: GO SUB 6ØØØ
```

Ejecuta el programa y como respuesta a los mensajes de pantalla teclea unos valores adecuados para D y para la posición del ojo. Esta vez la variable R corresponde al radio interno del tubo, mientras que el número de segmentos radiales está dado por N. Estos segmentos corresponden a las líneas longitudinales del globo. RT asigna la distancia (un radio) desde el origen al centro del tubo, mientras que N2 da el número de segmentos a lo largo de la circunferencia del tubo, que corresponden a las latitudes de la esfera. El valor de RT debe ser mayor o igual que R, aunque de hecho la rutina sigue funcionando bien aunque no ocurra esto. Modifica el valor de estas variables para observar el efecto resultante. Cuando RT es cero y N es igual que N2, el toro degenera en una esfera, tal como aparecía dibujada en la *rutina Globo*.

COMBINACIÓN DE IMÁGENES

En el programa anterior que dibujaba cuatro imágenes juntas en la pantalla, te resultará muy sencillo sustituir la *rutina Cubo* por la *rutina Globo* o la *rutina Toro*. De una forma igualmente sencilla, puedes utilizar una combinación cualquiera de las tres rutinas. Por ejemplo, podrías dibujar un cubo dentro de una esfera o de un toro, o incluso un cubo dentro de una esfera que a su vez está dentro de un toro. También puedes insertar sentencias GOTO en los puntos adecuados del programa para evitar que se dibujen las últimas caras de modo que la imagen no resulte muy recargada. Teclea las siguientes líneas y obtendrás una impresionante representación de una estación espacial:

```
 15Ø GO SUB 3ØØØ
3ØØØ LET R=6: LET N=24: LET
     RT=4Ø: LET N2=24: GO
     SUB 2ØØØ
3Ø3Ø FOR F=1 TO 6
3Ø4Ø LET A=(F+.25)*PI/3: LET
     L1=1Ø: LET L2=34: LET
     N1=12: LET R=2: LET
     N2=8: GO SUB 35ØØ
3Ø5Ø NEXT F
3Ø6Ø LET R=1Ø: LET N=12:
     LET RT=Ø: LET N2=12:
     GO SUB 2ØØØ
3Ø7Ø RETURN
35ØØ LET SA=SIN A
353Ø LET CA=COS A
354Ø LET T1=-SA: LET T2=CA:
     LET T3=Ø
355Ø LET T4=Ø: LET T5=Ø: LET
     T6=1
356Ø LET EA=(L2-L1)/N1
357Ø LET EB=L1
358Ø FOR E=Ø TO N1
359Ø LET T7=EB*CA: LET
     T8=EB*SA: LET T9=Ø
36ØØ LET XS=Ø: LET YS=Ø: LET
     N=N2: GO SUB 6ØØØ
361Ø LET EB=EB+EA
362Ø NEXT E
363Ø LET T1=CA: LET T2=SA:
     LET T3=Ø
364Ø FOR E=1 TO N2
365Ø LET EA=E*PI/8
366Ø LET T7=R*SA*COS EA:
     LET T8=-R*CA*COS EA:
     LET T9=R*SIN EA
367Ø LET XS=L1: LET YS=Ø:
     LET XE=L2: LET YE=Ø:
     GO SUB 95ØØ
368Ø NEXT E
369Ø RETURN
```

Este programa dibuja un toro con seis radios apuntando hacia el centro, en el que se encuentra una esfera. La imagen resultante se parece a determinados modelos de estación espacial.

El campo de experimentación que te permite el programa del toro es virtualmente ilimitado. Ensaya algunos valores pequeños de D (por ejemplo entre 100 y 500) junto con diversos valores de X, Y y Z para obtener interesantes vistas en perspectiva, especialmente al recortar el segmento más corto del toro. Prueba también con valores altos de D para observar las distorsiones que resultan.

Ahora ya es el momento de pasar a los toques finales, tales como el color del fondo y el de la tinta.

MEJORANDO LA CLARIDAD

Cuando las imágenes de los cuatro cubos son pequeñas, como sucede cuando los valores de X, Y y Z correspondientes a la posición del ojo son mucho mayores que la distancia de observación (D), es difícil ver lo que sucede cuando el programa avanza durante la fase de dibujo. Puedes mejorar la claridad insertando sentencias GOTO en lugares adecuados del programa entre las líneas 1000 y 1170. La más sencilla es GOTO 1220, insertada en la línea 1125 (que es una nueva línea) para el caso del programa de los cuatro cubos.

P y R

¿Hay alguna forma sencilla de acelerar el dibujo de la estación espacial?

La forma más radical de acelerar el programa consiste en reducir el número de pasos de las rutinas básicas: la *rutina Círculo* para la estación espacial y la *rutina Cubo* para el caso de los cubos. Sin embargo este método reducirá la suavidad de las elipses y estropeará la forma de las estructuras. Un método mejor es reducir el número de veces que se llama a las rutinas.

Podrías tener por ejemplo menos secciones transversales y longitudinales del toro, reduciendo los valores de R, N, RT y N2 en la rutina del toro que empieza en la línea 2000. Esto implica obviamente una modificación del programa para que la rutina sea llamada de forma que espacie con regularidad los círculos o las elipses y los saltos después de la última serie de curvas, en vez de seguir con los movimientos del dibujo más de lo necesario.

Probablemente la forma más sencilla de acelerar la ejecución consiste en saltarse el dibujo de determinadas secciones (tales como los radios de la estación). Esto se hace de forma sencilla con unos cuantos GOTO estratégicamente distribuidos.

A NUESTROS NUEVOS LECTORES

En las páginas centrales de la revista encontrarás la sección «Programación de juegos» que se compone de una serie de artículos coleccionables que continúan mes tras mes.

Como la paginación de estos artículos es siempre correlativa con la del mes anterior apreciarás que no se corresponde con la del resto de la revista, pudiendo parecer a los más despistados que faltan páginas o que se trata de un error de encuadernación.

ORDENACION, INDEXACION, BUSQUEDA E INSERCION

- BUSQUEDA E INCLUSION
- ORDENACION
- METODO DE SUSTITUCION
- METODO «BURBUJA»
- METODO «SHELL»

Si estás preparando, o tienes pensado hacerlo, algún programa de archivo, te será de gran ayuda conocer las distintas técnicas de ordenación, búsqueda e inserción de datos. Conociéndolas podrás aplicarlas, en cada caso, de la forma más adecuada, ahorrando tiempo y evitándote problemas.

Cuando se maneja una cantidad reducida de información podemos almacenarla, consultarla o alterarla sin grandes problemas sin preocuparnos mucho de utilizar unos procedimientos excesivamente rígidos. Puede bastarnos un lapicero y un papel para anotar nuestras necesidades de compra en unos grandes almacenes. Todo nuestro problema se resolverá anotando las cosas a medida que se nos ocurren y consultando la lista cada vez que pasemos por una sección donde vendan algo que recordemos vagamente que tenemos que comprar o exploremos nuestras notas para ver a qué planta debemos dirigirnos en cada momento. Por lo general no tendremos dificultades especiales aunque en nuestra lista tengamos la siguiente relación: bañador Luis, tazones desayuno, melones, harina, toallas pequeñas, dos paquetes café molido.

Si escribiésemos con la misma alegría la lista de las personas y teléfonos de una ciudad seguramente nos veríamos en aprietos a la hora de localizar a cualquiera de ellas. La compañía telefónica por lo menos así lo entiende y por ello nos ofrece dicha lista ordenada por apellidos, calles o profesiones (para el uso de su propio personal utiliza incluso otra ordenada por números de teléfono).

Para que una información de cierto volumen sea útil a la hora de ser ma-



nejada conviene que esté «ordenada» por algún criterio. Si no está ordenada puede servir, pero a costa de que el procedimiento de búsqueda sea más lento. Si las variaciones de una lista son numerosas y frecuentes, y el procedimiento de ordenación laborioso puede que no nos compense la ordenación si las consultas van a ser esporádicas. No es infrecuente que en el cuaderno que tenemos al lado del teléfono vayamos apuntando nombres y teléfonos uno a continuación de otro, y que los eliminemos con una simple tachadura cuando ya no nos valen. Este procedimiento nos obliga a coger la lista desde el principio y «pasear el dedo» hasta que encontremos lo que buscamos sin que haya gran diferencia con utilizar una agenda con clasificación alfabética. A la hora de añadir nombres, si la lista no es muy grande, pueden colocarse al final o «entre» renglones ya existentes y el procedimiento funciona.

Cuando se trata de localizar información sobre las diferentes partes de un libro existe también la posibilidad de «hojearlo» hasta encontrar la información que nos interese, aun cuando este procedimiento no es aconsejable si el libro es de un cierto tamaño o si las consultas son frecuentes. La solución adecuada es disponer de un «índice» donde se hallen clasificados por orden alfabético o temático los aspectos que nos interesen.

Esta necesidad de *ordenación, indexación, búsqueda* e *inclusión* se pone de manifiesto a la hora del manejo de bloques de información (ficheros) por los ordenadores. Todas estas operaciones pueden hacerse directamente en memoria central o incluso en el propio diskette (o disco duro). En este último caso podremos obviar las limitaciones de la memoria pero a costa de un alto precio en el desgaste del material y sobre todo en los tiempos de respuesta.

El manejo de datos en memoria central puede hacerse fundamentalmente con ayuda de variables subindicadas (numéricas o de caracteres), variables simples de caracteres y *disco RAM*. Nos vamos a ceñir a las primeras por ser las de uso más generalizado.

Cuando hablamos de variables subindicadas nos referimos a *vectores (listas), matrices de dos dimensiones (tablas)* y a *estructuras de más de tres dimensiones*, si bien estas últimas por lo general se descomponen por comodidad en matrices bidimensionales.

BÚSQUEDA E INCLUSIÓN

La búsqueda en una lista es «secuencial» cuando para encontrar un



determinado nombre (o valor numérico) se comienza por la primera posición y se van analizando todas hasta que se encuentra lo que buscamos, o hasta el final de la lista en el caso de que pueda haber más elementos que cumplan la condición impuesta (por ejemplo, todos los nombres que comiencen por MART). Este procedimiento es sólo adecuado cuando la lista está desordenada.

El mecanismo de búsqueda en una matriz A$( ) sería:

```
10 FOR i=1 TO n
20 IF A$(i)=" nombre buscado "
   THEN PRINT i
30 NEXT i
```

Los sucesivos valores de «i» nos indicarían los lugares de aparición del nombre buscado dentro de la lista.

Si disponemos de una lista ordenada podemos hacer una búsqueda más rápida mediante un sistema «dicotómico», «binario» o de «corte». Se toma el valor central de la lista y se compara con el que estamos buscando. Si éste es menor descartamos la segunda mitad de la lista y repetimos el proceso dividiendo por la mitad hasta que nos quedemos en un solo valor. Si este último valor es igual al que buscamos habremos hallado en qué lugar empieza en la lista y podremos hacer, a partir de este punto, una búsqueda secuencial de valores iguales hasta que aparezca un valor distinto. También pudiera ocurrir que el nombre que buscamos no exista, pero al menos sabríamos dónde debería aparecer de haber estado en la lista:

Una rutina para esta búsqueda puede ser:

```
 1 LET N=10:REM ELEGIR
   NUMERO DE ELEMENTOS A
   EXPLORAR
10 LET P =1:LET U=N LET
   R=U
20 LET R=INT((U-P)/2):LET
   M=P+R
30 IF X$<=A$(M) THEN LET
   P=P:LET U=M
40 IF X$>A$(M) THEN LET
   P=M+1:LET U=U
50 IF P<U THEN GO TO 20
60 IF X$=A$(M) THEN PRINT
   X$+" APARECE EN     ";M
70 IF X$<>A$(M) THEN PRINT
   X$+" NO EXISTE EN LA
   LISTA"
```

Este procedimiento de búsqueda es realmente rápido. El número de pasadas depende de la longitud de la lista.

Es normal que en algún momento se presente la necesidad de eliminar o añadir algún elemento de la lista. Con el procedimiento de búsqueda binaria sabremos dónde eliminar y proceder al correspondiente «corrimiento» para ocupar el espacio vacío e igualmente dónde colocar el nuevo valor, haciendo el «hueco» correspondiente. La ventaja radica en que el nuevo valor que añadamos queda ya ordenado.

ORDENACIÓN

Un ejemplo de conjunto desordenado sería un mazo de cartas «bien barajado».

Una lista decimos que está ordenada cuando sus elementos se hallan dispuestos uno a continuación de otro siguiendo un criterio, como puede ser el alfabético. Los ordenadores utilizan por lo general como referencia el código ASCII donde los números tienen valores inferiores a las letras mayúsculas y éstas a las minúsculas (prácticamente todos los manuales de ordenadores disponen de una tabla ASCII).

Existen varios métodos o procedimientos de ordenación («sorting» en la literatura inglesa). Cada uno de ellos tiene sus peculiaridades y una recomendación concreta. Unos son más fáciles de comprender y programar, otros son más rápidos a costa de utilizar más memoria RAM, otros, por el contrario, son parcos en consumo de memoria pero más lentos, etc. A continuación comentaremos algunos de ellos de forma que podamos sacar unas ideas claras que nos permitan ayudarnos a programar nuestras necesidades o juzgar algunas rutinas existentes. Al final figura un programa donde de una manera gráfica podemos experimentar con la ordenación.

MÉTODO DE SUSTITUCIÓN

Supongamos una lista («vector» o matriz de una dimensión) de «n» elementos.

El método consiste en ir comparando cada elemento con todos los que le siguen (1 con 2, 1 con 3, 1 con 4, etc., permutándolos si el segundo es mayor que el primero. Al final de la pasada el primer elemento será el de menor valor. En pasadas sucesivas se repite lo mismo pero desde el segundo lugar (2 con 3, 2 con 4, 2 con 5, etc.) y así sucesivamente. Una vez terminado el ciclo la lista quedará ordenada de menor a mayor.



La lista a ordenar se carga en la matriz A$( ) de dimensión «N» y la rutina de ordenación sería:

```
10 LET N=5:DIM A$(N,10)
20 FOR I=1 TO N:READ
   A$(I):NEXT I
30 FOR I=1 TO (N-1)
40 FOR H=I+1 TO N
50 IF A$(1)>A$(H) THEN LET
   C$=A$(I):LET
   A$(1)=A$(H):LET A$(H)=
   C$
60 NEXT H
70 NEXT I
100 FOR I=1 TO N:PRINT
    A$(I):NEXT I:STOP
110 DATA "CF","AA","DE",
    "SE","ZX"
```

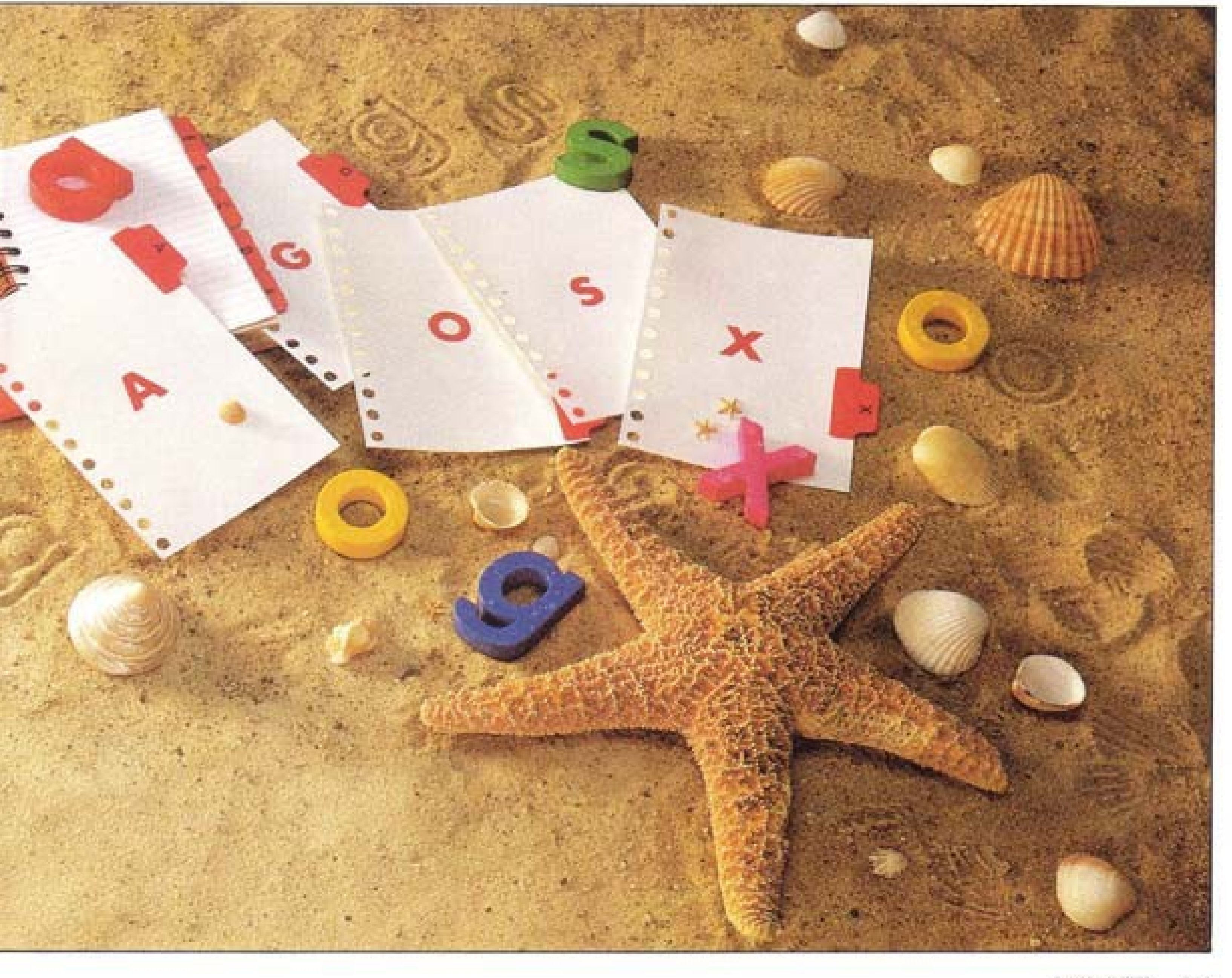
Éste sería el aspecto de las diversas pasadas del proceso:

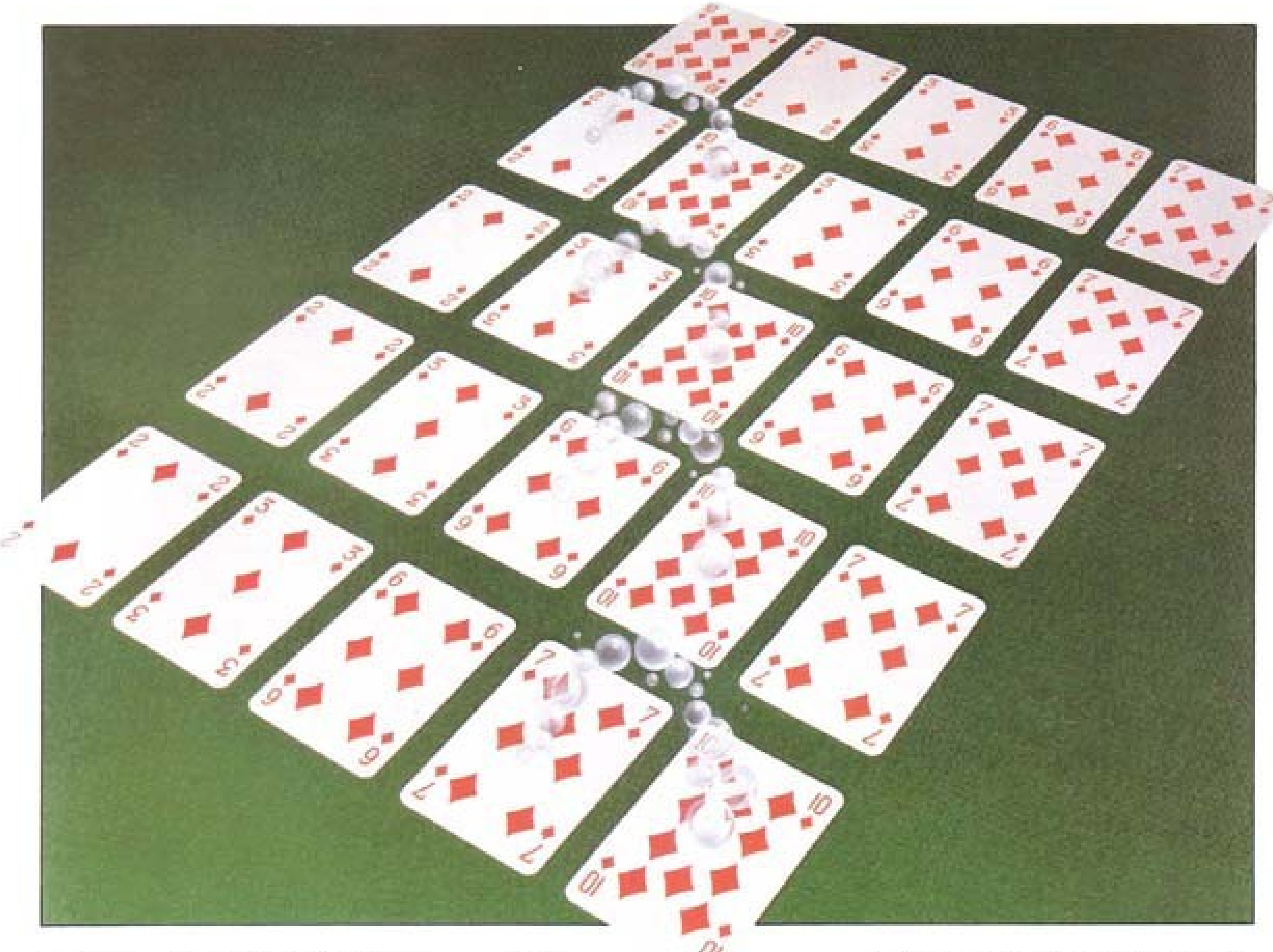
(CF)(AA)(AA)(AA) AA AA AA AA AA AA AA
(AA) CF CF CF (CF)(CF)(CF) CF CF CF CF
DE (DE) DE DE (DE) DE DE (DE)(DE) DE DE
SE SE (SE) SE SE (SE) SE (SE) SE (SE) SE
ZX ZX ZX (ZX) ZX ZX (ZX) ZX (ZX)(ZX) ZX

Este método es el más elemental e intuitivo, pero sin lugar a dudas el más lento. Cuando rodemos el programa de demostración del final veremos que para un número reducido de elementos puede ser suficientemente rápido. En este programa aparece el proceso a velocidad normal (muy rápido), ralentizado por el hecho de aparecer en pantalla los diversos pasos (rápido) y ralentizado a propósito para ver en «cámara lenta» el proceso (lento).

MÉTODO DE LA «BURBUJA»

Quizá es uno de los métodos más populares. Consiste en ir comparando parejas «sucesivas» de valores (1y2, 2y3, 3y4...9y10) e intercambiarlos si el primero es mayor que el segundo. Al final de la primera pasada el último valor será el mayor de la lista. La segunda pasada se hará por lo tanto para un valor menos y así sucesivamente. Si la ordenación se hace de mayor a me-





nor, las pasadas sucesivas irán «ascendiendo» los números menores como si se tratase de las burbujas de un líquido y de ahí su nombre.

Si al final de una de las pasadas no se ha producido la necesidad de intercambio de valores quiere decir que la lista entera está ordenada y se puede suspender el proceso con el consiguiente ahorro de tiempo. Ésta es una de las ventajas principales. Su uso está indicado por tanto en aquellos casos en que la lista es grande pero está casi ordenada (por ejemplo al añadir un elemento a una lista ordenada).

```
10 LET N=5:DIM A$(N,10)
20 FOR I=1 TO N:READ
   A$(I):NEXT I
30 FOR I=1 TO (N-1):LET k=0
40 FOR H=1 TO (N-I)
50 IFA$(H)>A$(H+1)THENLET
   C$=A$(H):LET
   A$(H)=A$(H+1):LET
   A$(H+1)=C$:LETk=k+1
60 NEXT H:IF k=0 THEN LET
   I=N-1
70 NEXT I
100 FOR I=1 TO N:PRINT
    A$(I):NEXT I:STOP
110 DATA "CF", "AA", "DE",
    "SE", "ZX"
```

Éste sería el aspecto de las diversas pasadas del proceso

```
(CF) AA AA AA (AA) AA AA AA
(AA)(CF) CF CF (CF)(CF) CF CF
DE (DE)(DE) DE DE (DE)(DE) DE
SE SE (SE)(SE) SE SE (SE)(SE)
ZX ZX ZX (ZX) ZX ZX ZX (ZX)
```

MÉTODO «SHELL»

Existen variantes de estos métodos como puede ser el denominado «Shell», en donde el aumento de velocidad es considerable. En este caso se toma como referencia la mitad de la lista y se van haciendo comparaciones entre parejas separadas el valor de la referencia (el primero con el mitad, el segundo con el mitad más uno, etc.). Al final de la primera pasada la lista habrá quedado ordenada de una manera «alternada». Variando la referencia al valor entero de la mitad de la referencia anterior en formas sucesivas se llega a una ordenación bastante más rápida, aunque realmente no se puede apreciar su justo valor más que en código máquina. En Basic la lentitud de los bucles enmascara sus ventajas.

Para el mismo caso anterior:

```
10 LET N=10:LET D=N: DIM
   A$(N,10)
20 FOR I=1 TO N:READ
   A$(I):NEXT I
30 IF D<=1 THEN GO TO 100
40 LET D=INT(D/2):LET F=N-D
50 LET Z=0
60 FOR H=1 TO F
70 IF A$(H)>A$(H+D) THEN
   LET C$=A$(H):LET
   A$(H)=A$(H+D):LET
   A$(H+D)=C$:LET Z=1
```

```
80 NEXT H
90 IF Z>0 THEN GO TO 50
95 IF Z<=0 THEN GO TO 30
100 FOR I=1 TO N: PRINT
    A$(I):NEXT I: STOP
110 DATA "SERT", "DERT",
    "CFGT", "ZSDL", "AASER",
    "XSD", "DE", "SERTY",
    "FF", "AZZW"
```

MÉTODO «SHELL-METZNER»

Se trata de una variante del método «Shell» con el cual se consigue una mayor velocidad:

Para el mismo caso anterior:

```
10 LET N=10:DIM A$(N,10)
20 FOR I=1 TO N:READ
   A$(I):NEXT I
30 LET D=INT(N/2)
40 IF D<1 THEN GO TO 100
50 LET F=N-D
60 FOR I=1 TO F
70 FOR H=I TO 1 STEP -D
80 IF A$(H+d)<A$(H) THEN
   LET C$=A$(H):LET A$(H)
   =A$(H+D):LET A$(H+D)=
   C$
90 NEXT H:NEXT I
95 LET D=INT(D/2):GOTO 40
100 FOR I=1 TO N:PRINT
    A$(I):NEXT I:STOP
110 DATA "SERT", "DERT",
    "CFGT", "ZSDL", "AASER",
    "XSD", "DE", "SERTY",
    "FF", "AZZW"
```

Como curiosidad, sobre todo si no disponemos de ordenador, podemos experimentar los métodos anteriores con ayuda de las cartas de una baraja. Elegiremos por ejemplo cinco cartas cualquiera y las colocaremos verticalmente u horizontalmente y luego las iremos moviendo e intercambiando de acuerdo con las indicaciones de la rutina correspondiente.

ORDENACIÓN RÁPIDA

Hasta ahora hemos considerado sólo la posibilidad de reordenar la propia matriz, sin ningún gasto adicional de memoria.

Si nos encontramos con un cajón lleno de fichas que hemos de clasificar, como método alternativo a la reclasificación en el propio cajón podemos seguir el procedimiento de tomar otro cajón vacío e ir pasando fichas del primero al segundo pero dejándolas ya cada una en su lugar. El proceso es evidentemente más rápido y no hay que dar tantas vueltas a las fichas.

Con un ordenador el proceso es análogo. Necesitaremos crear otra matriz auxiliar de capacidad adecuada (si la lista es grande podemos tener problemas de memoria) a donde iremos pasando una a una las posiciones de la primera matriz ayudándonos de una búsqueda binaria para hacer la inclusión en el lugar oportuno.

En el caso de listas con varios campos en cada registro o tablas (matrices de dos dimensiones) puede surgirnos la necesidad de ordenarlas según diversos criterios (apellidos, profesiones, calles, número de cliente, etc.). Bastaría con hacer tantos procesos de ordenación como fuera preciso tal como se ha visto en el apartado anterior. El único problema sería el elevado volumen de información redundante que tendríamos que manejar.

Lo normal en este caso sería conservar la lista (matriz) inicial y confeccionar tantos «índices» como sea preciso. Cada índice estaría compuesto por una referencia inequívoca (no valdría un apellido, ya que puede haber más de un cliente con el mismo nombre. Sí valdría el D.N.I. o el número de cliente) y el contenido del campo respectivo, y hecha la ordenación correspondiente. Por ejemplo, el número de referencia de cliente y los apellidos, o el número de teléfono, o la ciudad, etc. Uno de los índices sería también el número de cliente asociado al número de orden que ocupe ese cliente en la tabla básica.

Cuando deseemos saber el número de teléfono de un cliente en particular iríamos primero a la matriz de «índice apellido» y buscaríamos qué número de cliente le corresponde. A continuación consultaríamos el «índice número de cliente», el cual nos informaría dónde encontrar la ficha completa de dicho cliente, incluido su teléfono.

Esta forma de proceder, aunque un poco laboriosa, es la única que nos permite manejar un gran volumen de información con un consumo razonable de memoria.

Las Bases de datos hacen uso de estos principios manejando informaciones almacenadas en ficheros residentes en diskettes o discos duros con los correspondientes pasos por memoria central.

Lo normal es que los índices se vayan actualizando automáticamente a medida que se añade o elimina información de la base de datos.

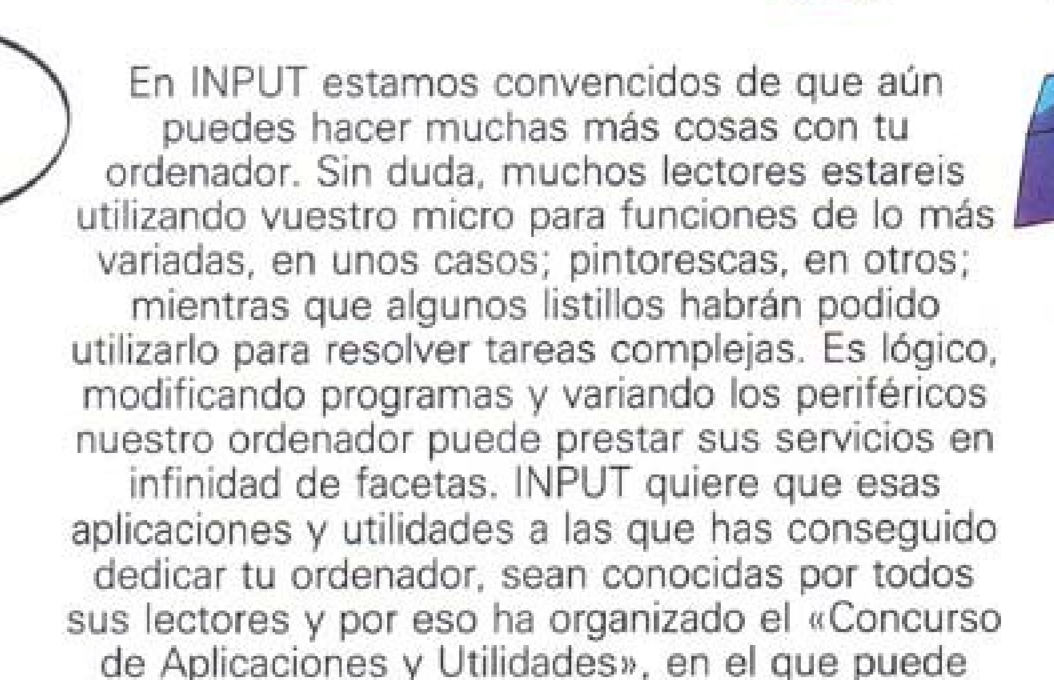
Como hemos comentado, incluimos un programa que muestra el funcionamiento de los distintos algoritmos de ordenación. El objetivo del mismo es, sobre todo, ser didáctico. En la pantalla se muestra paso a paso el proceso de ordenación de una tabla de 10 elementos. Tecléalo, experimenta y saca tus conclusiones sobre las ventajas de utilizar uno u otro método.

```
20 BORDER 6: PAPER 6: INK 0:
   CLS
30 DIM A$(10,5): DIM B$(10,5):
   LET C$="     ":LET
   T$=C$+"     ": LET
   Q$=T$+" "
100 REM FORMACION DE LA
    LISTA
120 PRINT AT 17,0;
    "INTRODUZCA DIEZ
    NOMBRES"
130 FOR I=1 TO 10: INPUT
    "NOMBRE ? (max. 5c.) ";
    A$(I): LET B$(I)=A$(I):
140 PRINT AT I,0,I;TAB (4);
    A$(I): NEXT I: BEEP 1,10
160 CLS: PRINT "OPCIONES
    METODOS ORDENACION":
    PRINT : PRINT
170 PRINT "1
    SUSTITUCION..(LENTO)"
180 PRINT "2
    SUSTITUCION..(RAPIDO)"
190 PRINT "3
    SUSTITUCION..(MUY
    RAPIDO)"
200 PRINT "4
    BURBUJA......(LENTO)"
210 PRINT "5
```

```
     BURBUJA......(RAPIDO)"
220  PRINT "6
     BURBUJA......(MUY
     RAPIDO)"
230  PRINT : PRINT : PRINT
     "PULSE OPCION"
240  LET P$=INKEY$: IF P$=""
     OR P$>"6" THEN GO TO
     240
250  LET P=VAL (P$): IF P=1 OR
     P=2 THEN GO SUB 300
255  IF P=3 THEN GO SUB 350
260  IF P=4 OR P=5 THEN GO
     SUB 400
265  IF P=6 THEN GO SUB 500
270  GO TO 160
300  CLS : GO SUB 2000: GO
     SUB 2020: GO SUB 820
310  FOR N=1 TO 9: FOR
     H=N+1 TO 10
315  GO SUB 1000: GO SUB
     2020: GO SUB 760
320  IF A$(N)<=A$(H) THEN GO
     SUB 990: GO TO 330
325  IF P=1 THEN GO SUB 700:
     LET S$=A$(N): LET
     A$(N)=A$(H): LET
     A$(H)=S$
326  IF P=2 THEN LET
     S$=A$(N): LET
     A$(N)=A$(H): LET
     A$(H)=S$
330  GO SUB 990: GO SUB
     2020: GO SUB 1000
340  NEXT H: NEXT N: GO SUB
     2020: GO SUB 830: PRINT
     AT 19,0;"CUALQUIER
     TECLA PARA SEGUIR"
345  IF INKEY$="" THEN GO TO
     345
346  RETURN
350  CLS : GO SUB 2000: GO
     SUB 820
360  FOR N=1 TO 9: FOR
     H=N+1 TO 10
370  IF A$(N)>A$(H) THEN LET
     S$=A$(N): LET
     A$(N)=A$(H): LET
     A$(H)=S$
380  NEXT H: NEXT N: GO SUB
     2020: GO SUB 830: GO
     SUB 800: PRINT
     "CUALQUIER TECLA PARA
     SEGUIR"
390  IF INKEY$="" THEN GO TO
     390
395  RETURN
400  CLS: GO SUB 2000: GO
     SUB 2020: GO SUB 820:
     LET R=9
410  LET S=0: FOR N=1 TO R:
     LET H=N+1
415  GO SUB 1000: GO SUB
     2020: GO SUB 760
420  IF A$(N)<A$(H) THEN GO
     SUB 990: GO TO 430
425  IF P=4 THEN GO SUB 700:
     LET S$=A$(N): LET
     A$(N)=A$(H): LET
     A$(H)=S$: LET S=S+1
426  IF P=5 THEN LET
     S$=A$(N): LET
     A$(N)=A$(H): LET
     A$(H)=S$: LET S=S+1
430  GO SUB 990: GO SUB
     2020: GO SUB 1000
440  NEXT N: LET R=R-1: IF
     R=0 OR S<>0 THEN GO
     TO 410
444  GO SUB 2020: GO SUB
     830: GO SUB 800: PRINT
     "CUALQUIER TECLA PARA
     SEGUIR"
445  IF INKEY$="" THEN GO TO
     445
446  RETURN
500  CLS: GO SUB 2000: GO
     SUB 820: LET R=9
510  LET S=0: FOR N=1 TO R:
     LET H=N+1
520  IF A$(N)>A$(H) THEN LET
     S$=A$(N): LET
     A$(N)=A$(H): LET
     A$(H)=S$: LET S=S+1
540  NEXT N: LET R=R-1: IF
     R=0 OR S<>0 THEN GO
     TO 510
544  GO SUB 2020: GO SUB
     830: GO SUB 800: PRINT
     "CUALQUIER TECLA PARA
     SEGUIR"
545  IF INKEY$="" THEN GO TO
     545
546  RETURN
700  GO SUB 990: REM CAMBIO
710  PRINT AT H,18;C$;TAB
     (25); A$(H): GO SUB 990
720  PRINT AT H,18; A$(N); TAB
     (25); C$
730  PRINT AT N,18;C$;TAB
     (25); A$(H): GO SUB 990
740  PRINT AT N,18; A$(H); TAB
     (25);C$: RETURN
760  PRINT AT N,12; "----> ";
     A$(N); AT H,12; "----> ";
     A$(H): RETURN
820  PRINT AT 12,3; "L.ORIG
     PROC. DE ORDENACION":
     RETURN
830  PRINT AT 12,3; "L. ORIG
     L.ORD        ": RETURN
990  REM RETARDO 40
995  IF P=1 OR P=4 THEN
     PAUSE 40
998  IF P=2 OR P=5 THEN
     PAUSE 1
999  RETURN
1000 REM BORRADO PROCESO
     DE ORDENACION
1010 FOR I=1 TO 10: PRINT AT
     1,18; T$: NEXT I: RETURN
2000 REM LISTA ORIGINAL
2010 FOR I=1 TO 10: LET
     A$(I)=B$(I): PRINT AT I,0,
     I; TAB (4); B$(I): NEXT I:
     RETURN
2020 REM LISTA A CLASIFICAR
2030 FOR I=1 TO 10: PRINT AT
     I,12; A$(I): NEXT I:
     RETURN
```

Si se te hace difícil encontrar INPUT en tu kiosco habitual, resérvalo por adelantado, o háznoslo saber para que podamos remediarlo





En INPUT estamos convencidos de que aún puedes hacer muchas más cosas con tu ordenador. Sin duda, muchos lectores estareis utilizando vuestro micro para funciones de lo más variadas, en unos casos; pintorescas, en otros; mientras que algunos listillos habrán podido utilizarlo para resolver tareas complejas. Es lógico, modificando programas y variando los periféricos nuestro ordenador puede prestar sus servicios en infinidad de facetas. INPUT quiere que esas aplicaciones y utilidades a las que has conseguido dedicar tu ordenador, sean conocidas por todos sus lectores y por eso ha organizado el «Concurso de Aplicaciones y Utilidades», en el que puede participar cualquiera de nuestros lectores.





UTILIDADES Y APLICACIONES: Si tu ordenador controla la calefacción de tu casa, gobierna un robot, dirige un pequeño negocio, organiza la maqueta de tu tren eléctrico, o cualquier cosa interesante u original; envíanos información gráfica y listados de tus programas, grabados en un cassette, diskette o microdrive.

Todo ello habrá de venir acompañado por un texto que aclare cuál es su objetivo, el modo de funcionamiento y una explicación del cometido que cumplen las distintas rutinas que lo componen. El texto se presentará en papel de tamaño folio y mecanografiado a dos espacios. No importa que la redacción no sea muy clara y cuidada; nuestro equipo de expertos se encargará de proporcionarle la forma más atractiva posible.

UN JURADO propio decidirá en cada momento qué colaboraciones reúnen los requisitos adecuados para su publicación, y evaluará la cuantía del premio en metálico al que se hagan acreedoras.

No olvideis indicar claramente para qué ordenador está preparado el material, así como vuestro nombre y dirección y, cuando sea posible, un teléfono de contacto. Entre todos los trabajos recibidos durante los próximos tres meses SORTEAREMOS:

- **Un premio de 50.000 ptas.**
- **Un premio de 25.000 ptas.**
- **Un premio de 10.000 ptas.**
  en material microinformático a elegir por los afortunados.

¡No os desanimeis!, por muy simples o complejas que puedan parecer vuestras ideas, todas están revisadas con el máximo interés.

INPUT SINCLAIR
Aribau, 185. Planta 1.ª
08021 BARCELONA

NOTA: INPUT no se responsabiliza de la devolución del material que no vaya acompañado por un sobre adecuado con el franqueo correspondiente.

REFINA TUS GRAFICOS DE PANTALLA

Una vez que dominas los rudimentos del dibujo de imágenes en pantalla, puedes empezar a ampliar el campo de tus esfuerzos artísticos, utilizando algunos de los comandos gráficos especializados de que dispone tu ordenador. Comandos tales como MOVE, PLOT, DRAW y CIRCLE, te permiten dar rienda suelta a tu imaginación, creando cualquier tipo de imágenes, desde un gráfico de tipo comercial hasta el escenario de fondo de una emocionante aventura.

En artículos anteriores ya vimos la manera de utilizar los sencillos comandos de dibujo para crear en la pantalla imágenes a base de líneas y, en determinados casos, añadirle el color. Pero tú puedes ampliar estas técnicas con nuevos métodos para dibujar puntos, líneas, triángulos, cuadrados y círculos. Dichos métodos, utilizados ellos solos o en combinación con los colores, se convierten en un poderoso medio de generación de imágenes estáticas o incluso en movimiento.

CIRCULOS Y ARCOS

Los círculos y los arcos figuran entre las «herramientas» más útiles para dibujar gráficos estáticos de pantalla con un Spectrum.

El siguiente programa del partido de golf muestra la manera de utilizarlos para dibujar árboles, cercas, agua, hoyas y arenales, a la vez que permite describir los principios que rigen este tipo de representaciones.

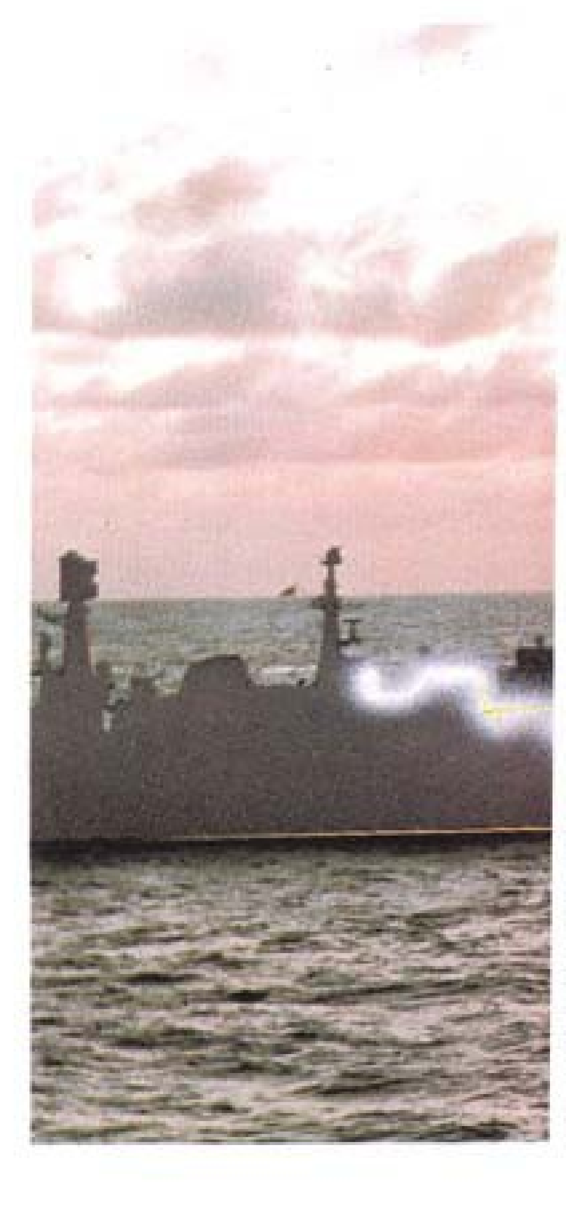
Puedes comprobar tus progresos a medida que vas avanzando, ejecutando cada grupo de líneas. No hagas un NEW en tu máquina después de cada ejecución; si dejas las líneas intactas, terminarás con la escena de la figura 1. No obstante, antes de empezar con los círculos, es mejor que saques el chalet del camino. Empieza pues tecleando las siguientes líneas:

```
 9Ø BORDER 4: PAPER 4:
    CLS
2ØØ LET W=1Ø: LET S=5Ø
3ØØ PRINT INK Ø;AT 2,2;" ";AT
    2,4;" ";AT 2,6;" "
23Ø DRAW INK 2;S,Ø
24Ø LET W=W+2: LET S=S-4
25Ø NEXT C
26Ø FOR B=148 TO 162
27Ø PLOT INK 2;1Ø,B
28Ø DRAW INK 2;5Ø,Ø
29Ø NEXT B
295 DRAW INK 2;1Ø,-3: DRAW
    Ø,-11
3ØØ PRINT INK Ø;AT 2,2;" "AT
    2,4;" ";AT 2,6;" "
```

Las líneas 200 a 250 dibujan el tejado. Empieza en el punto 10, 162 de la pantalla, con una línea de 50 pixels de ancha. A continuación su anchura disminuye en 4 pixels por cada pixel de subida. Un bucle semejante que se extiende entre las líneas 260 y 290, se ocupa de dibujar las paredes con dos líneas adicionales (línea 295) para formar un porche. A continuación la línea 300 se encarga de las ventanas de la forma más simple posible, imprimiendo por tres veces en las paredes un cuadrado negro tomado de los caracteres gráficos de ROM.

La forma más sencilla de dibujar un círculo completo en el Spectrum es utilizar un comando CIRCLE. Pero si sólo quieres tener una parte de un círculo, la forma más sencilla de conseguirlo es utilizar además una sentencia DRAW convencional. Variando esta sentencia puedes conseguir una cantidad enorme de efectos, por lo que conviene que experimentes un poco antes de seguir más adelante. Ensaya por ejemplo:

```
1Ø PLOT 13Ø,3Ø
2Ø DRAW Ø,1Ø,1
3Ø GOTO 2Ø
```

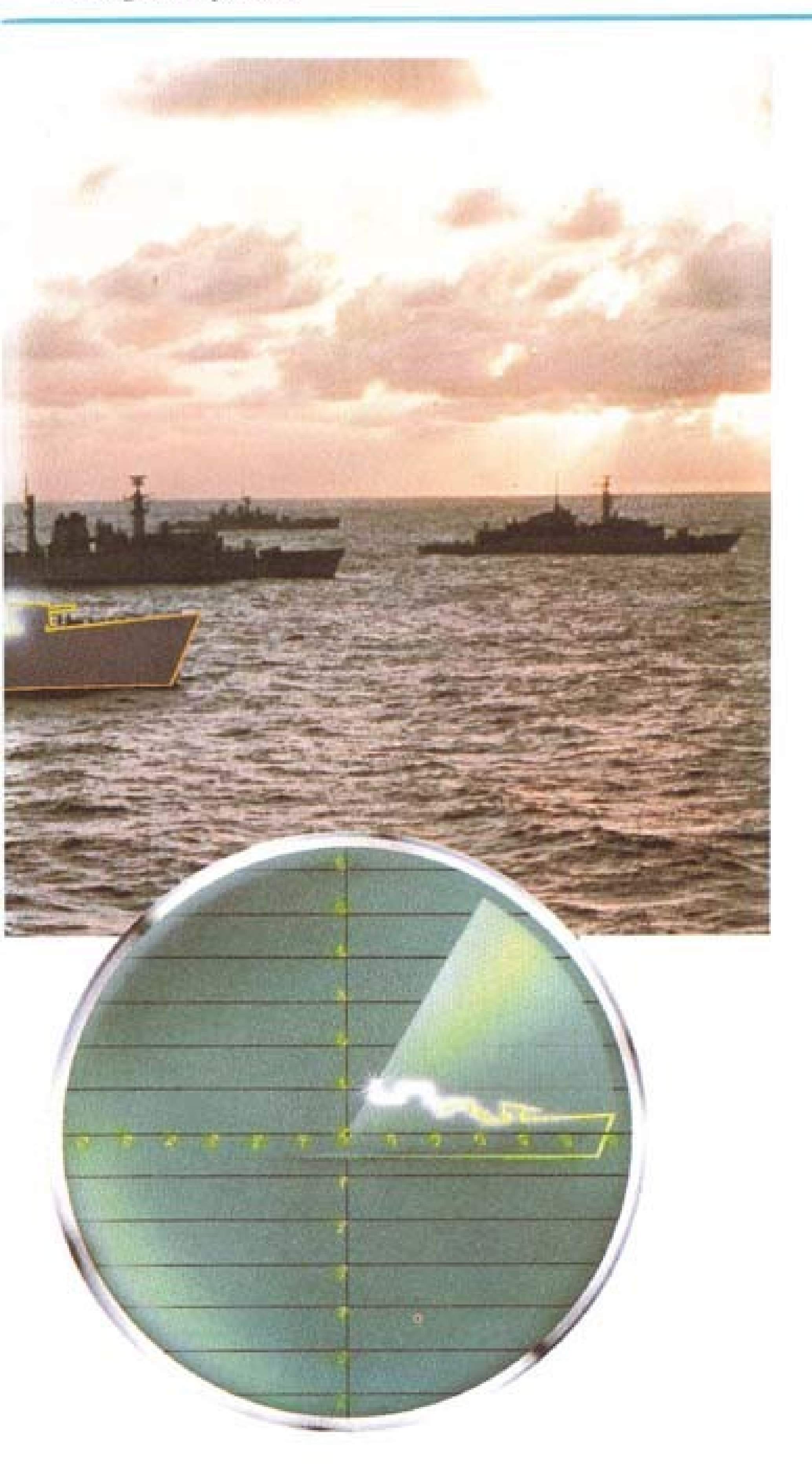


(No te preocupes por el mensaje de error).

Como te habrás dado cuenta, los dos primeros números de la línea 20, dicen al ordenador que dibuje una línea a partir del punto recién dibujado y extendiéndose hasta llegar a un punto situado 10 pixels más arriba. El último número hace que esta línea sea curva, en vez de ir en línea recta. El radio de curvatura obtenido depende de lo grande o pequeño que sea el número. Este número indica al Spectrum que dibuje parcialmente un círculo. El círculo completo está representado

Los ordenadores domésticos ofrecen muchas posibilidades para el artista en ciernes. Aquí tienes unas cuantas sugerencias para ampliar el uso de los comandos gráficos del BASIC y crear nuevas imágenes de pantalla.

- ESCENAS DE LA PARTIDA DE GOLF
- IMAGENES CON CIRCULOS Y ARCOS
- ARBOLES Y ARBUSTOS



por el producto de 2 por π, con lo cual el número 1 hace que se dibuje aproximadamente la sexta parte de un círculo (exactamente se dibuja la fracción $1/2\pi$ del círculo). Si intentas cambiar la línea 20 por la siguiente:

```
20 DRAW 0,10,2
```

te encontrarás con una serie de curvas más pronunciadas. Análogamente, con 0, 10, 3 obtendrás una figura en diente de sierra, con 0, 10, 4 tendrás parte de un cierre con eslabones de cadena (o medio tronco de palmera, según cómo lo mires), mientras que con 0, 10, 6 te resultará una espiral. Si por el contrario tecleas

```
20 DRAW 0,10,2*PI
```

puede que te lleves una sorpresa. Lo que el Spectrum pretende hacer es dibujar un círculo con dos de sus puntos en línea recta. Pero como dicho círculo no cabría en tu cuarto de estar —en realidad no cabría en todo el Sistema Solar— sólo dibuja una parte del mismo. Para dibujar un paisaje, ensaya lo siguiente:

```
10 PLOT 0,100
20 DRAW RND*5+5,0,2
30 GO TO 20
```

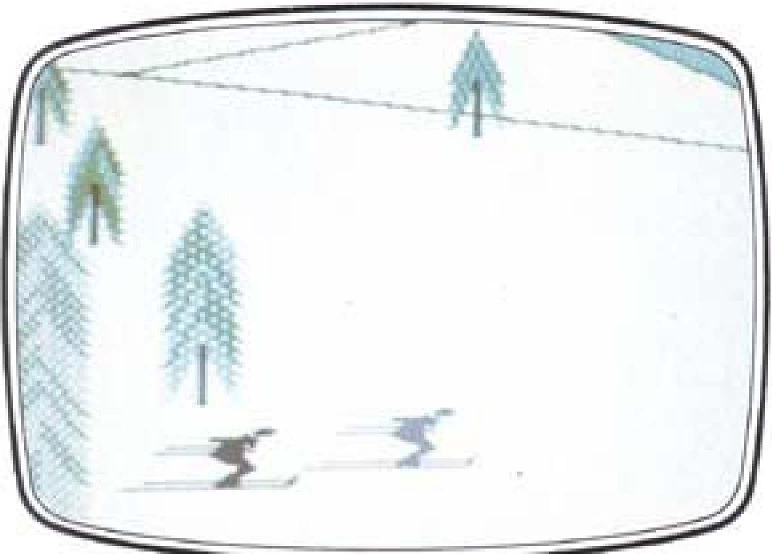
Con esto conseguirás dibujar el oleaje de un mar agitado. Para tener aguas más tranquilas, ensaya RND*10 + 10, o incluso RND*15 + 10.

Un punto importante que debes recordar en relación con estos arcos, es que un número negativo al final de tu línea DRAW hará que se dibuje la imagen especular del correspondiente arco. Borra ahora las líneas 10 a 30 y estarás en condiciones de construir la escena de la partida de golf.

LA CERCA Y EL LAGO

El programa de la partida de golf contiene dos ejemplos de arcos utilizados





Utilización de arcos y círculos para la partida de golf.

como efectos gráficos; el primero es el de la pequeña cerca que hay enfrente del chalet; el segundo es el lago.

Primeramente teclea estas líneas:

```
31Ø FOR F=Ø TO 84 STEP 3
32Ø PLOT F,142
33Ø DRAW 3,Ø,-3
34Ø NEXT F
345 DRAW 35, -42: DRAW
    -12,-6
```

El punto de partida sobre la pantalla es en este caso 0,142. Lo que hace la línea 330 es dibujar una serie de arcos muy poco curvos —en realidad son semicírculos de sólo tres pixels de anchura— para formar la cerca. El número total está gobernado por el bucle FOR ... NEXT.

A continuación, teclea y ejecuta las siguientes líneas:

```
1ØØ LET X=13Ø: LET Y=125:
    LET Z=5Ø
11Ø PLOT INK 5;X,Ø
12Ø DRAW INK 5;Y,Z,-1.25
13Ø LET X=X+1: LET Y=Y-1:
    LET Z=Z-1
14Ø IF X>254 THEN GO TO 17Ø
15Ø IF Z<1 THEN LET Z=Ø
16Ø GO TO 11Ø
```

Esta figura es ya más complicada. En primer lugar el ordenador dibuja un punto situado a 130 pixels desde la izquierda y a 0 pixels desde el fondo de la pantalla. A continuación traza una línea hasta el punto situado 125 pixels a la derecha y 50 pixels hacia arriba, contando desde el fondo de la pantalla y «doblando» la línea −1,25 radianes a medida que la va trazando.

A partir de este punto, son las variables las que dominan. La variable *x* inicia cada línea un pixel más a la derecha y la variable *y* hace que cada línea sea un pixel más corta que la anterior (en caso contrario se saldría de la pantalla), mientras que la variable *z* hace que cada línea termine un pixel más abajo que la anterior.

Evidentemente, llegará un momento en que *z* llegará a ser un número negativo, haciendo que el ordenador intente (sin conseguirlo) imprimir algo por debajo del fondo de la pantalla. De aquí la necesidad de la línea 150, que hace que todas las líneas cortas próximas al final del programa terminen en la línea de fondo de la pantalla.

ARBOLES Y ARBUSTOS

El programa de la partida de golf utiliza también círculos completos, como sustitutivos de la sentencia PAINT y otras análogas que figuran en el repertorio de otros ordenadores pero no en el del Spectrum. Los círculos se emplean para producir árboles y matojos. Las siguientes líneas te permitirán tener unos cuantos arbustos aleatoriamente distribuidos detrás del «verde»:

```
4ØØ FOR R=172 TO 168 STEP -1
41Ø LET X=RND*45+195
415 PLOT X,R-2: DRAW Ø,-2
42Ø CIRCLE X,R,RND*2+1
44Ø CIRCLE X+1Ø,R,RND*2+1
45Ø NEXT R
```

En cambio estas otras líneas harán que aparezca una maleza análoga en el lado derecho:

```
46Ø FOR R=135 TO 172 STEP 6
47Ø LET Y=252
48Ø CIRCLE Y,R,RND+2
49Ø NEXT R
```

Los árboles de la esquina inferior izquierda son demasiado grandes para ser dibujados aleatoriamente. Por eso en este caso hemos utilizado una técnica distinta, empleando las sentencias READ ... DATA:

```
 9ØØ FOR W=1 TO 3
 91Ø READ A,B
 92Ø PLOT A,B
 93Ø DRAW Ø,-24
 94Ø LET F=RND*5+5
 95Ø CIRCLE A-1Ø,B+F,F:
     CIRCLE A,B+F,F: CIRCLE
     A+1Ø,B+F,F
 96Ø CIRCLE A-5,B+F*2,F:
     CIRCLE A+5,B+F*2,F
 97Ø CIRCLE A,B+F*3,F
 98Ø NEXT W
3ØØØ DATA 2Ø,7Ø,52,85,84,1ØØ
```

El truco consiste en dibujar primero los troncos, avanzando hacia abajo a partir de los puntos originales de la sentencia PLOT, de forma que no aparezcan marcas ciegas de troncos a través del «follaje». El dibujo de los troncos se hace por medio de 20, 70, etc., además de la línea de DATA 3000. La línea 940 sirve para aleatorizar el tamaño del follaje, mientras que el B + F que figura en la línea 950 sirve para asegurarse de que las filas del fondo de los círculos empiezan a una distancia adecuada por encima de los troncos.

LOS TOQUES FINALES

La «tee» de lanzamiento se dibuja por medio de las siguientes líneas:

```
1000 LET T=30
1010 FOR Y=0 TO 10
1020 PLOT T,Y
1030 DRAW -30,30
1040 LET T=T+2
1050 NEXT Y
```

Y estas otras líneas sirven para dibujar las banderas en los verdes:

```
170 PLOT 220,140
180 DRAW 0,15: DRAW 8,-3:
    DRAW -8,-2
190 PLOT 22,120
195 DRAW 0,18: DRAW 9,-3:
    DRAW -9,-2
```

Por último necesitarás algunos «bunkers» u hoyos de arena. Cuando no se





dispone de una sentencia PAINT, la manera más inmediata de dibujarlos, aunque muy lenta, es empezar con una pequeña elipse y hacer que vaya creciendo pixel a pixel hasta que alcance un tamaño razonable.

La forma de dibujar elipses se explicará en detalle en un artículo posterior que dedicaremos a las funciones matemáticas. Mientras tanto, puedes introducir las siguientes líneas. Cuando las ejecutes puedes irte tranquilamente a tomar un café para darle tiempo a que acabe:

```
1495 LET R=1
15ØØ FOR X=Ø TO 2*PI STEP PI/
     18Ø
151Ø PLOT INK 6;168+R*SIN
     X,147+R*COS X/2.5
152Ø PLOT INK 6;235+R*SIN
     X,1Ø6+R*COS X/2.75
153Ø PLOT INK 6;225+R*SIN
     X,97+R*COS X/2.5
154Ø NEXT X
155Ø LET R=R+2
156Ø IF R>2Ø THEN GO TO
     6ØØØ
157Ø GO TO 15ØØ
```

También tienes otra alternativa más rápida y sencilla, aunque los dibujos resultantes son más bastos. Consiste en utilizar tus GDUs para construir los bunkers.

DOMINANDO EL TABLERO (I)

Se levanta el telón para la presentación del Juego OTELO. Programa este sencillo juego de estrategia y de engaño y desafía a tu ordenador. Pero, ¡cuidado!, no es tan sencillo como parece.

OTELO es un juego de estrategia que se juega sobre un tablero de ocho por ocho casillas —un tablero de ajedrez o de damas puede servirnos—. Las reglas son muy simples y el juego cuenta con varios trucos también.

El objetivo consiste en capturar o *comer* el mayor número de fichas posibles a tu oponente. Cada jugador ha de colocar una ficha en el tablero hasta llenarlo por completo. Al comenzar cada uno cuenta con dos fichas y ha de intentar capturar las del jugador contrario *rodeándoselas*. Ello se logra colocando una ficha extra al final de una fila de fichas, de manera que el adversario se vea acorralado por tus fichas.

Todas las fichas adversarias que hayas acorralado serán reemplazadas ahora por fichas tuyas.

El número de puntos será simplemente el número de fichas de cada jugador que haya sobre el tablero durante cada jugada. El ganador será aquel que consiga tener el mayor número de piezas sobre el tablero cuando éste esté lleno.

En esta versión, tú juegas contra el ordenador, el cual también muestra en pantalla el tablero y lleva la puntuación.

CONSEJOS Y TRUCOS

Al igual que en cualquier otro juego de estrategia, éste también cuenta con algunos trucos que pueden serte de gran ayuda.

Si ésta es la primera vez que juegas al OTELO, los siguientes consejos te serán muy útiles.

Las fichas de los extremos son muy valiosas, pues no pueden ser recuperadas una vez han sido capturadas —la razón de ello es debido a que estas fichas no pueden ser acorraladas como las de otras posiciones del tablero—. Como consecuencia, pueden ser muy significativas para ganar, y es muy importante capturar las esquinas, incluso sacrificando un movimiento que podría habernos proporcionado una mayor puntuación. Asimismo, también son intocables cada una de las fichas emplazadas junto a las fichas de las esquinas.

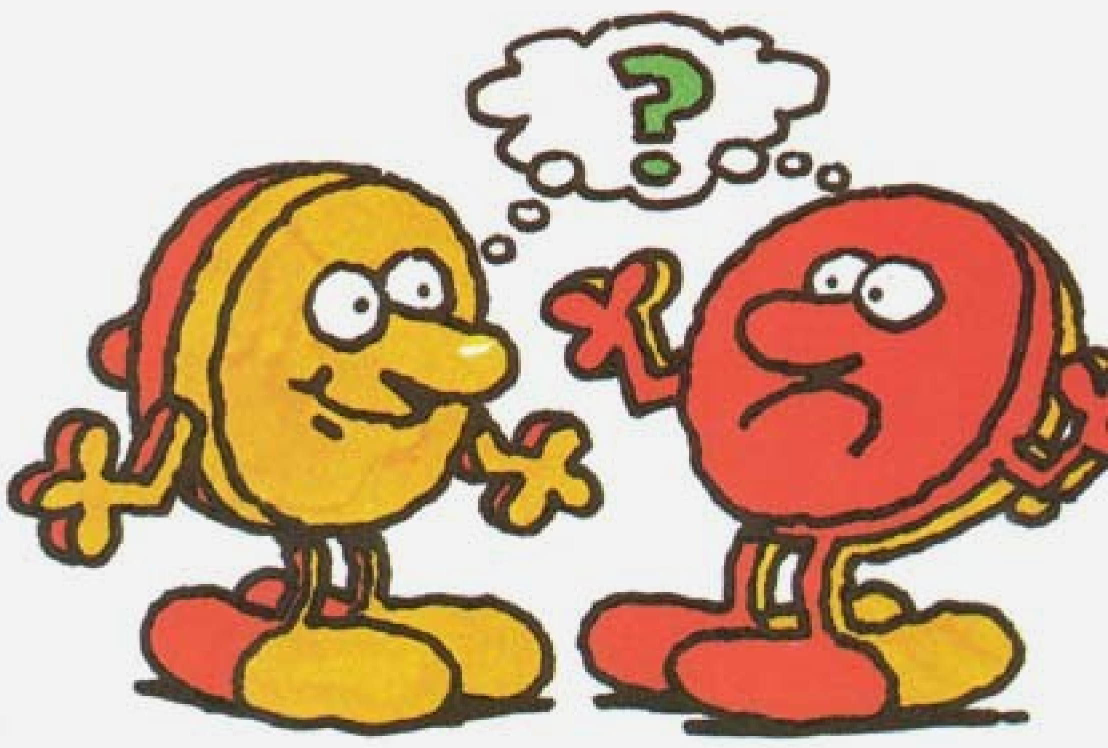
Puesto que una pieza puede enlazar más de una línea —arriba, abajo y en diagonal—, el movimiento más obvio no siempre será el mejor, mientras que en las últimas fases del juego, a menudo tú puedes enlazar dos o tres líneas añadiendo una sola ficha.

Debes pensar siempre con anticipación al adversario. Tal vez puedas conseguir engañar a tu oponente creando situaciones favorables para ti —para conquistar posiciones vitales— simulando una mala movida de tus fichas.

EL PROGRAMA

El programa juega el papel de tu oponente —con fichas negras—, y ya verás cómo un programa, comparativamente sencillo, es capaz de jugar con mucha destreza el desafiante juego OTELO. Una de las grandes ventajas que presenta el programa de ordenador OTELO, en comparación a un juego normal de tablero, es que en la versión de ordenador te ahorras toda la parte del trabajo difícil. El ordenador te evita el tener que voltear las fichas o sustituirlas por otras de diferente color. Tu único trabajo consiste en concentrarte en el juego.

Puesto que el ordenador considera todas las posibilidades, tarda un poco en realizar sus movimientos de fichas, aunque se vuelve más veloz a medida que avanza el juego y quedan cada vez menos casillas vacías.



JUGANDO AL OTELO

Cuando ejecutes el programa, el ordenador te preguntará si deseas mover tú primero. Cada vez que hagas un movimiento tendrás que introducir dos coordenadas. Éstas indicarán tu posición dentro de una escala entre uno y ocho —los números de líneas y de columnas se editan de arriba abajo a un lado del tablero. Las coordenadas se introducen indicando primero la posición horizontal (línea) y luego la vertical (columna).

El programa no reconoce a un compañero de juego poco hábil, ni tampoco es capaz de adivinar si estás aburrido, así pues, introduciendo un 0 como coordenada, se da fin al juego.

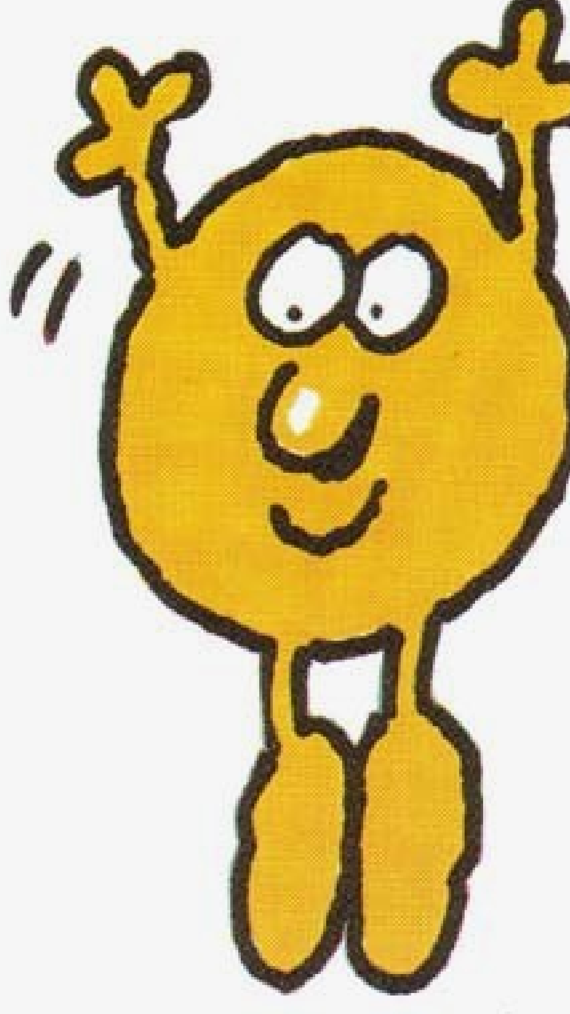
PRÓLOGO

A continuación, teclea la primera sección del juego OTELO. Si ejecutas con RUN el programa en esta fase, verás la primera pantalla y los gráficos, pero no podrás jugar todavía. No olvides grabar (SAVE) el programa, de manera que puedas añadirle luego la segunda sección.

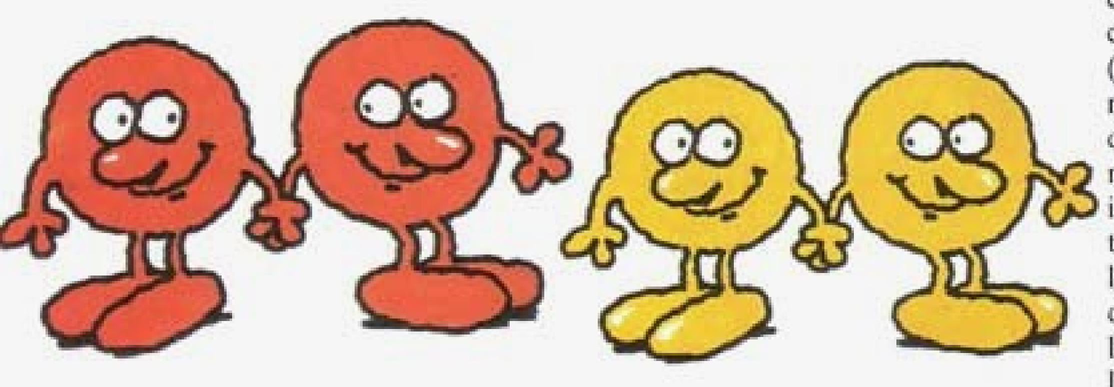
```
10 BORDER 0: PAPER 7: INK 1:
   CLS
15 PRINT AT 1,11; INVERSE 1;
   "OTELO"
20 PRINT AT 10,0;"QUIERES
   EMPEZAR TU ('S' O 'N')?":
   INPUT X$: IF X$="" THEN
   GO TO 20
30 LET X$=X$(1): IF X$<>"S"
   AND X$<>"N" AND
   X$<>"s" AND X$<>"n"
   THEN GO TO 20
40 LET CP=1: IF X$="N" OR
   X$="n" THEN LET CP=2
100 DIM B(8,8): DIM C(8): DIM
    D(8,2): DIM X(60): DIM
    Y(60): DIM N(60)
110 LET B(4,4)=1: LET B(4,
    5)=2: LET B(5,4)=2: LET
    B(5,5)=1
120 FOR F=1 TO 8: READ A:
    LET D(F,1)=A: READ A: LET
    D(F,2)=A: NEXT F
130 DATA -1,-1,0,-1,1,-1,
    -1,0,1,0,-1,1,0,1,1,1
140 FOR F=0 TO 7: READ A,B,
    C: POKE USR "A"+F,A:
    POKE USR "B"+F,B: POKE
    USR "C"+F,C: NEXT F
150 DATA 204,0,0,51,60,60,
    204,126,66,51,126,66,
    204,126,66,51,126,66,
    204,60,60,51,0,0
```

Las líneas 10 a 40 se encargan de la primera pantalla que ve el jugador. La línea 10 determina los colores y borra la pantalla. El título del juego se edita con la línea 15. La línea 20 edita (PRINT) un símbolo indicador de *listo*, y te pregunta si deseas comenzar tú primero. La respuesta se registra en X$. Seguidamente, en la línea 30, X$ queda reducida a su primera letra. CP es el jugador actual y toma el valor 1 para el humano, y el valor 2 para el ordenador. La línea 40 verifica la respuesta «S» o «N».

Las líneas 100 a 130 inicializan las variables y los vectores necesarios en el juego. La línea 100 se encarga de los vectores. B(x,y) representa el tablero y los valores registrados en cada elemento representan el estado de la correspondiente casilla del tablero —si un elemento es cero, la casilla está vacía; si es uno, es que está ocupada por una ficha perteneciente al jugador, y si el valor es dos, la casilla está ocupada por una ficha del ordenador. C(x) se usa para comprobar el movimiento del jugador, D(x,y) contiene los desplazamientos X e Y para las ocho posibles direcciones de movimiento. X(x), Y(x) y N(x) se utilizan para calcular el movimiento del ordenador.



La línea 110 determina las posiciones iniciales sobre el tablero. Cada jugador tiene colocadas dos fichas en el centro del tablero. Las posibles direcciones de esta posición se leen (READ) de las líneas DATA en la línea 130. Cada número representa un desplazamiento X o Y, siendo los números negativos movimientos hacia la izquierda o hacia la parte superior del tablero. Las líneas 140 y 150 colocan los UDG para las casillas vacías y las dos diferentes fichas. La línea 140 lee los datos (DATA) de la línea 150 y coloca los UDG A, B y C.



PRIMER ACTO: EL BUCLE PRINCIPAL

```
500 GO SUB 1000
505 IF CS+PS=64 THEN GO TO
    4000
510 LET EG=0: IF CP=1 THEN
    GO SUB 2000: GO SUB
    1000: IF EG=1 THEN GO
    TO 4000
515 IF CS+PS=64 THEN GO TO
    4000
520 IF CP=2 THEN GO SUB
    3000
530 GO TO 500
```

El bucle principal del programa se ejecuta desde la línea 500 a la línea 530. La línea 500 llama a la subrutina que traza el tablero. Las líneas 505 y 515 verifican si el tablero está lleno. Lo realizan comprobando si CS (puntuación del ordenador) y PS (puntuación del humano) suman 64.

La línea 510 coloca a cero el *flag* de fin de juego (EG). La sección media de la línea llama a la subrutina *turno para el jugador*, seguida de la rutina *mostrar tablero*. Si, una vez ejecutadas estas subrutinas, ha sido colocado EG, el programa salta a la rutina *fin del juego*, comenzando en la línea 4000. El movimiento del ordenador es llevado a cabo por la línea 520 si CP es 2.

SEGUNDO ACTO: TRAZANDO EL TABLERO

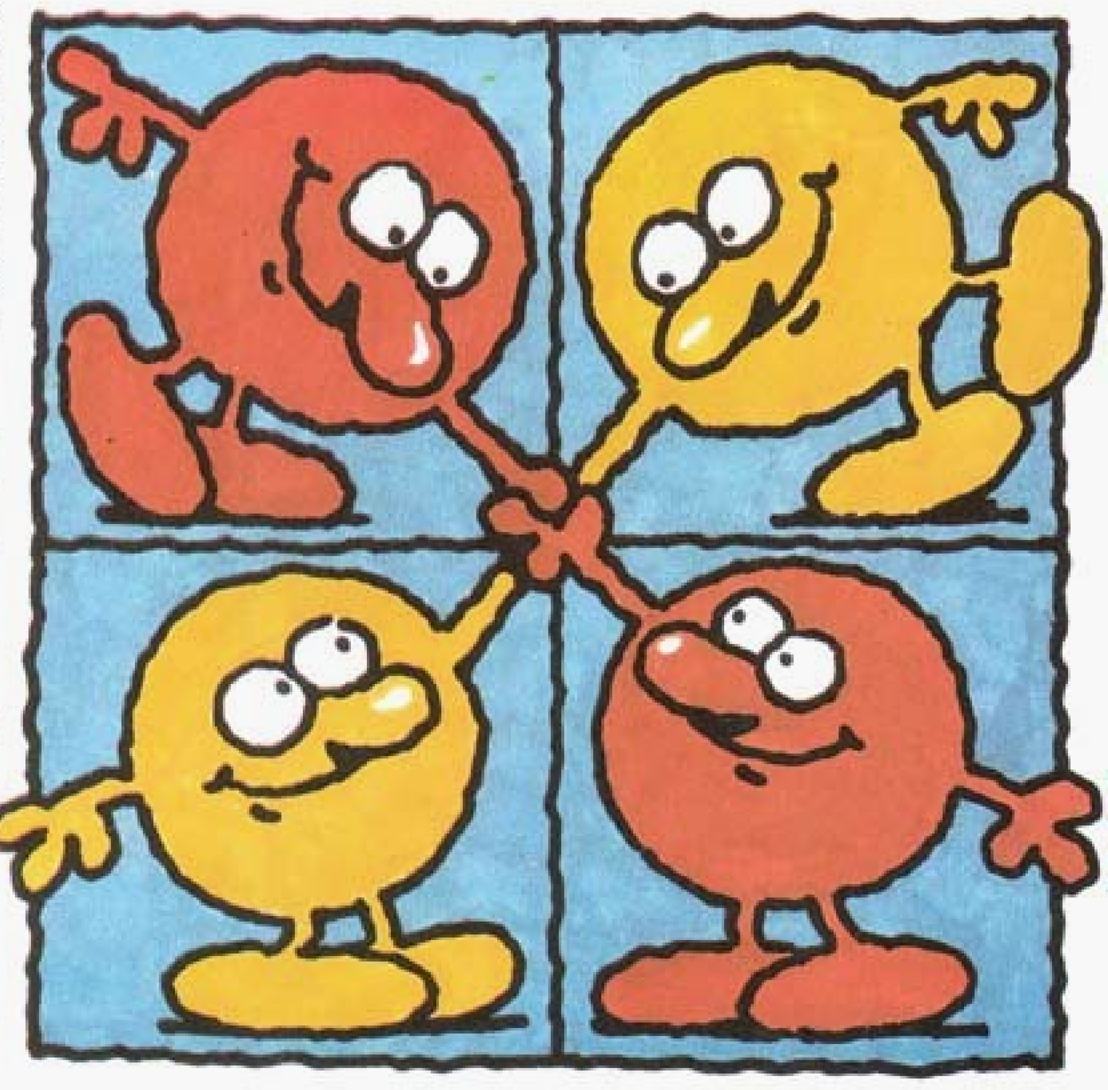
```
1000 CLS : PRINT TAB 11;
     "12345678": LET PS=0:
     LET CS=0
1010 FOR F=1 TO 8: PRINT TAB
     9;F;" ";: FOR G=1 TO 8
1020 IF B(F,G)=0 THEN PRINT
     CHR$ 144;
1030 IF B(F,G)=1 THEN PRINT
     INK 2;CHR$ 145;: LET
     PS=PS+1
1040 IF B(F,G)=2 THEN PRINT
     INK 2;CHR$ 146;: LET
     CS=CS+1
1050 NEXT G: PRINT : NEXT F
1052 LET P$="PUNTOS": IF
     PS=1 THEN LET
     P$="PUNTO"
1054 LET Q$="PUNTOS": IF
     CS=1 THEN LET
     Q$="PUNTO"
1060 PRINT '' INK 2;
     "JUGADOR=";TAB 22;
     "ORDENADOR=": PRINT
     PS;" ";P$;TAB 22;CS;" ";
     Q$
1070 RETURN
```

El tablero lo dibuja la rutina que se inicia a partir de la línea 1000. La pantalla se borra, se editan los números de las columnas, y los marcadores se colocan a cero.

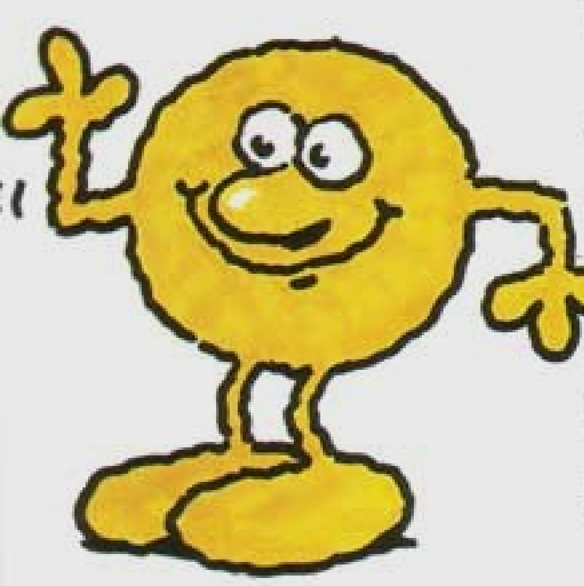
El bucle FOR...NEXT entre las líneas 1010 y 1050 edita el tablero en la pantalla, línea a línea. A medida que se van mostrando en pantalla las fichas, se van incrementando los puntos en el marcador. Las líneas 1052 y 1054 van sumando los puntos editados (PRINT) en la línea 1060.

TERCER ACTO: EL MOVIMIENTO DEL JUGADOR

```
2000 PRINT AT 14,0;"CUAL ES
     TU MOVIMIENTO (FIL,
     COL)?": INPUT X,Y
2005 IF X=0 THEN LET EG=1:
     RETURN
2006 IF X=9 THEN LET CP=2:
     RETURN
2010 IF X<1 OR X>8 OR Y<1
     OR Y>8 THEN GO TO
     2000
2020 IF B(X,Y)=0 THEN GO TO
     2070
2040 PRINT AT 17,0;"NO
     PUEDES MOVER A UNA
     CASILLA YA OCUPADA":
     FOR F=1 TO 500: NEXT F
```



```
2050 PRINT AT 17,0;"
     GO TO 2000
2070 LET NF=0: FOR F=1 TO
     8: LET CF=0: IF X+D(F,
     1)=0 OR X+D(F,1)=9
     THEN GO TO 2075
2071 IF Y+D(F,2)=0 OR Y+
     D(F,2)=9 THEN GO TC
     2075
2072 IF B(X+D(F,1),Y+D(F,
     2))=2 THEN LET CF=1:
     LET NF=1
2075 LET C(F)=0: IF CF=1
     THEN LET C(F)=F
2080 NEXT F
2090 STOP
```



El movimiento del jugador se introduce en las líneas 2000 a 2270, pero en este caso, el programa sólo llega hasta la línea 2090.

Después de un símbolo indicador, las coordenadas (X e Y) son introducidas en la línea 2000. La línea 2005 comprueba si X es igual a cero, y coloca el *flag* EG.

La entrada del jugador se verifica como errónea en la línea 2010 y, en caso necesario, retrocede saltando a la línea 2000.

La casilla elegida es sometida a verificación con el fin de comprobar si está vacía, y si así es, el programa salta a la línea 2070.

Si la casilla ya está ocupada, la línea 2060 edita un mensaje de error.

El último par de líneas —2060 y 2070— de esta sección del programa comprueba si la nueva ficha ha sido colocada cerca de una ficha del ordenador.

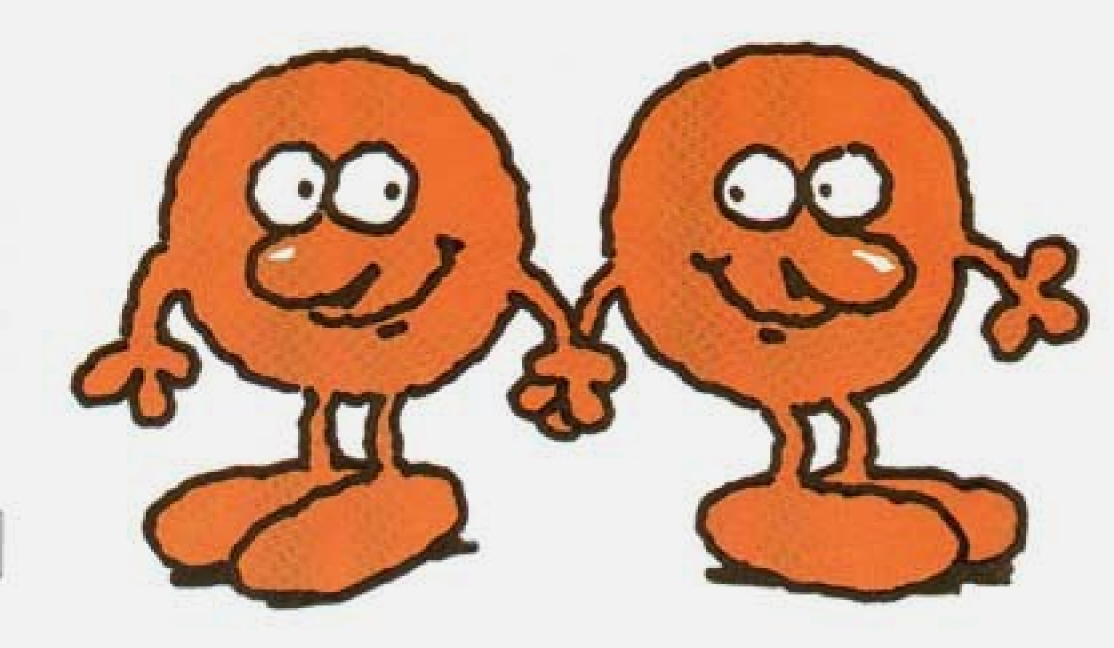
DOMINANDO EL TABLERO (y II)

- SEGUNDA PARTE DE OTELO
- COMPLETANDO EL TURNO DEL JUGADOR
- EL ORDENADOR MUEVE
- ANUNCIANDO EL GANADOR

Después de un breve descanso, pasemos a presenciar el nudo y desenlace final de OTELO.

A continuación, vamos a efectuar algunos movimientos astutos.

Cuando hayas completado esta segunda parte del juego OTELO tendrás ante ti todo el programa para poner a prueba tu ingenio. He aquí las líneas necesarias para completar el turno del jugador, junto con la rutina de movimiento del ordenador y el final de la rutina de juego.



DESPUÉS DEL DESCANSO

```
2090 IF NF=1 THEN GO TO
     2120
2100 PRINT AT 17,0;"TU
     MOVIMIENTO NO SIGUE A
     UNA DE MIS FICHAS":
     FOR F=1 TO 500: NEXT F
2110 PRINT AT 17,0;"       ":
     GO TO 2000
2120 LET RF=0: FOR Q=1 TO
     8: IF C(Q)=0 THEN GO TO
     2170
2130 LET XP=X: LET YP=Y
2140 LET XP=XP+D(Q,1): LET
     YP=YP+D(Q,2): IF XP=0
     OR XP=9 OR YP=0 OR
     YP=9 THEN LET C(Q)=0:
     GO TO 2170
2145 IF B(XP,YP)=2 THEN GO
     TO 2140
2150 IF B(XP,YP)=1 THEN LET
     RF=1: GO TO 2170
2160 IF B(XP,YP)=0 THEN LET
     C(Q)=0
2170 NEXT Q
2180 IF RF=1 THEN GO TO
     2210
2190 PRINT AT 17,0;"TU
     MOVIMIENTO NO
     FLANQUEA UNA FILA":
     FOR F=1 TO 500: NEXT F
2200 PRINT AT 17,0;"       ":
     GO TO 2000
2210 FOR Q=1 TO 8: IF C(Q)=0
     THEN GO TO 2250
2220 LET XP=X+D(Q,1): LET
     YP=Y+D(Q,2)
2230 IF B(XP,YP)=1 THEN GO
     TO 2250
2240 LET B(XP,YP)=1: LET
     XP=XP+D(Q,1): LET
     YP=YP+D(Q,2): GO TO
     2230
2250 NEXT Q
2260 LET B(X,Y)=1
2270 LET CP=2: RETURN
```

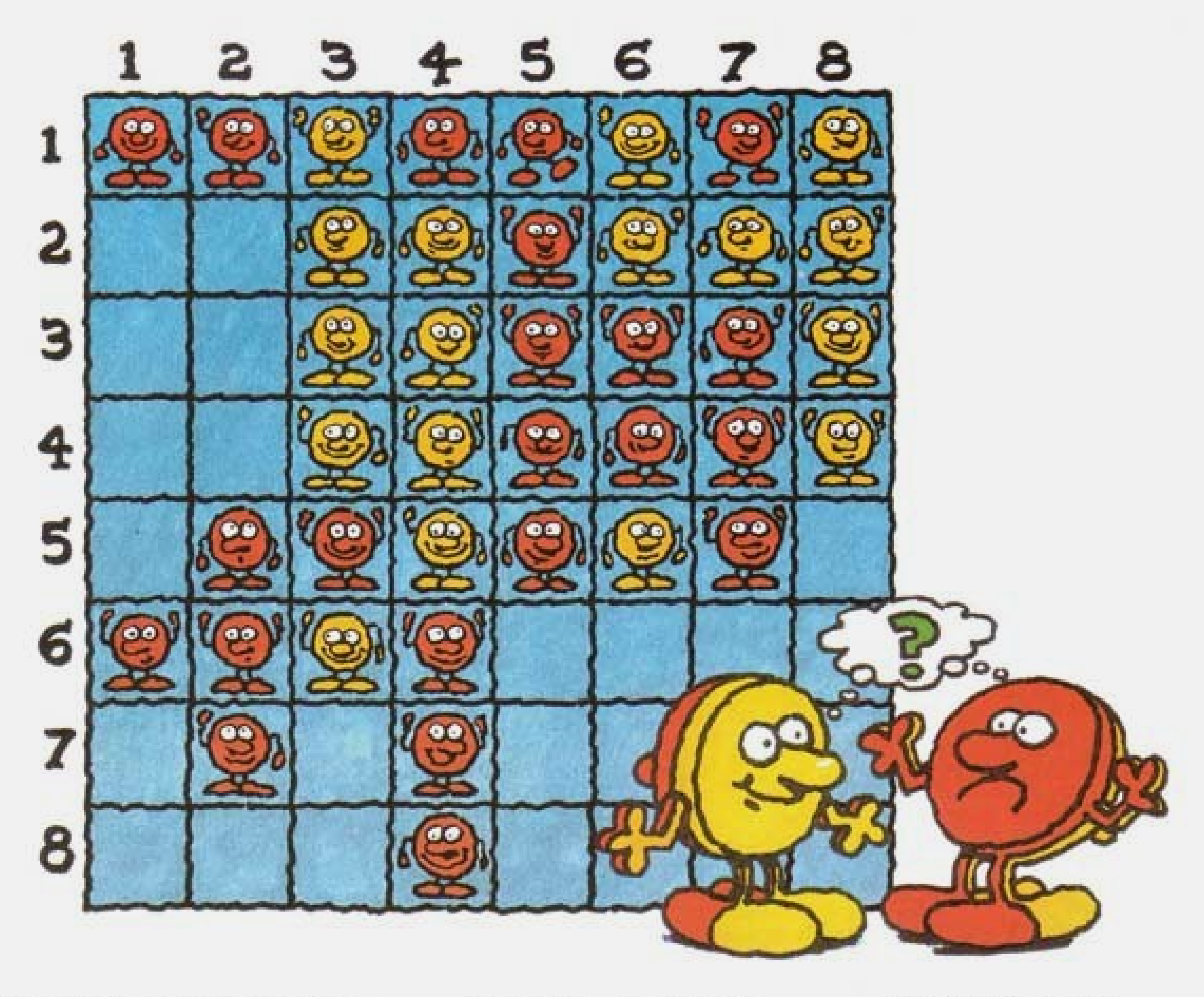
La línea 2090 comprueba el *flag* NF, para verificar antes de saltar a la línea 2120, si hay una ficha perteneciente al ordenador en una casilla adyacente. Si no hay ninguna ficha, la línea 2100 edita un mensaje de error.

A continuación, el programa comprueba en las líneas 2120 a 2170 si el movimiento cierra una línea o fila del tablero. La línea 2140 asegura que la posición que ha sido comprobada entra dentro de los límites del tablero. Si no es así, se abandona esta dirección y se prueba con la siguiente. La línea 2145 asegura que la ficha comprobada pertenece al ordenador. Si es así, el programa retrocede a la línea 2140 para actualizar la posición.

En la línea 2180 se verifica el éxito de la última operación. Si se ha encontrado una fila, el programa salta a la línea 2210. Las líneas 2190 a 2200 editan un mensaje para comunicar al jugador que no ha sido cerrada una línea, y el programa retrocede a la línea 2000.

El movimiento en sí se efectúa en las líneas 2210 a 2260. El bucle de la línea 2210 comprueba si se ha localizado una fila adecuada, verificando C(Q). Si no hay ninguna fila en esa dirección, el programa salta al siguiente NEXT Q en la línea 2250. XP e YP se colocan en la primera casilla de la fila tomada —ver línea 2040.

En la línea 2230, el ordenador verifica el final de una línea. Si se encuentra un final de línea, el programa salta a la línea 2250. La línea 2240 coloca la casilla a uno, luego salta a la



línea 2230 para comprobar la siguiente casilla.

En la línea 2260, la casilla del jugador se coloca a uno. El *flag* CP se coloca a dos para el ordenador en la línea 2270 y el programa hace un RETURN.

EL MOVIMIENTO DEL ORDENADOR

```
3ØØØ PRINT ''"PENSANDO...":
     LET NF=1: LET MX=Ø:
     FOR X=1 TO 8: FOR Y=1
     TO 8
3Ø1Ø IF B(X,Y)<>Ø THEN GO TO
     3Ø7Ø
3Ø2Ø FOR F=1 TO 8: LET XP=X:
     LET YP=Y: LET DX=D(F,
     1): LET DY=D(F,2): LET
     RF=Ø
3Ø3Ø LET XP=XP+DY: LET
     YP=YP+DX: IF XP=Ø OR
     XP=9 OR YP=Ø OR YP=9
     THEN GO TO 3Ø6Ø
3Ø4Ø IF B(XP,YP)=1 THEN LET
     RF=1: GO TO 3Ø3Ø
3Ø5Ø IF B(XP,YP)=2 AND RF=1
     THEN LET N(NF)=F: LET
     X(NF)=X: LET Y(NF)=Y:
     LET NF=NF+1: LET F=9
3Ø6Ø NEXT F
3Ø7Ø NEXT Y: NEXT X: LET
     NF=NF-1
3Ø75 IF NF=Ø THEN GO TO
     33ØØ
3Ø8Ø FOR F=1 TO NF: LET
     X=X(F): LET Y=Y(F): LET
     DX=D(N(F),1): LET
     DY=D(N(F),2): LET CF=Ø
3Ø9Ø LET X=X+DY: LET
     Y=Y+DX: IF B(X,Y)=1
     THEN LET CF=CF+1: GO
     TO 3Ø9Ø
31ØØ IF CF>MX THEN LET
     MX=CF: LET MF=F
311Ø NEXT F
318Ø FOR F=1 TO 8: LET
     X=X(MF): LET Y=Y(MF):
     LET DX=D(F,1): LET
     DY=D(F,2)
319Ø LET X=X+DY: LET
     Y=Y+DX
3195 IF X<1 OR X>8 OR Y<1
     OR Y>8 THEN GO TO
     326Ø
32ØØ IF B(X,Y)=1 THEN GO TO
```

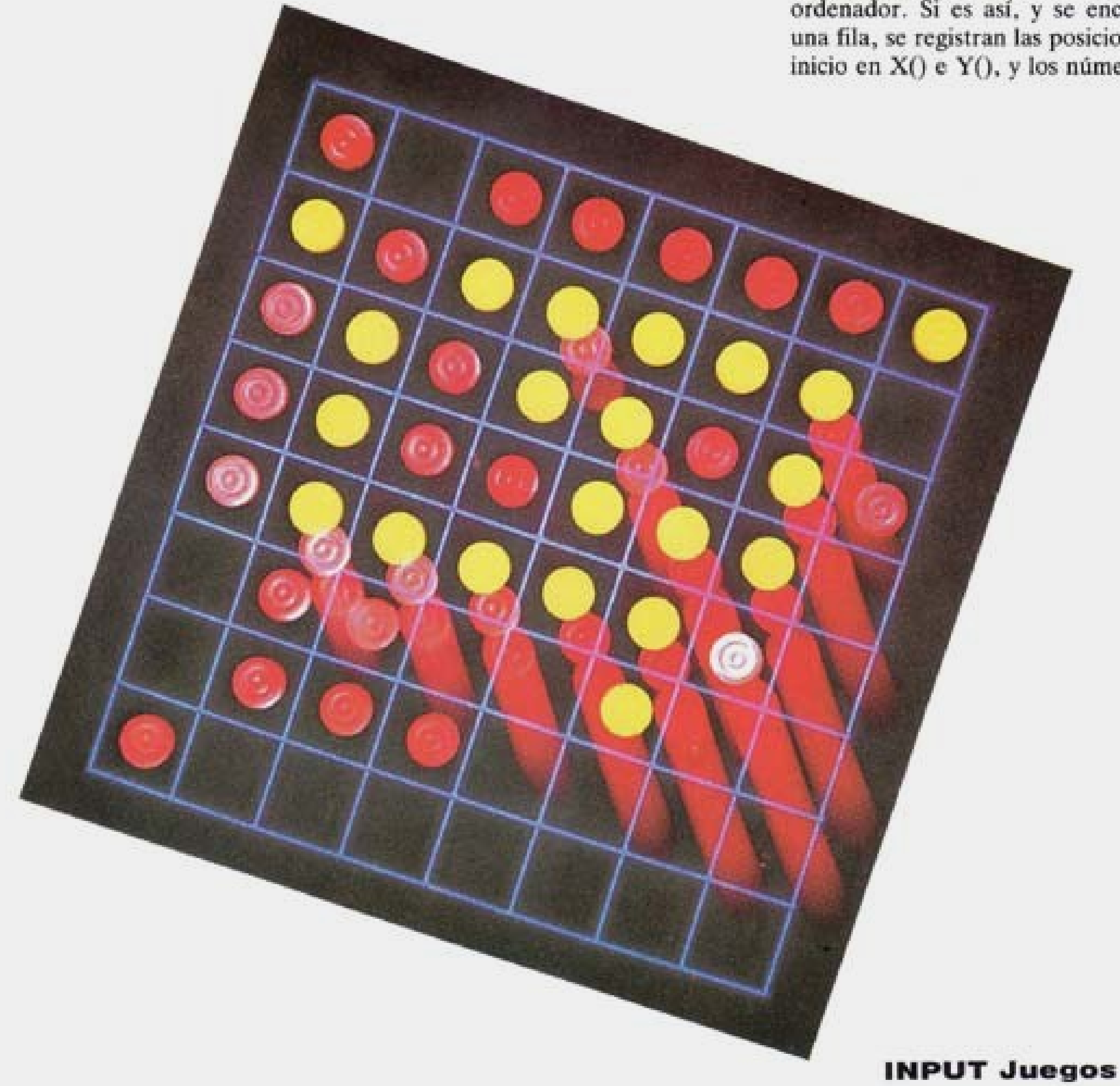
```
     3190
3210 IF B(X,Y)=2 THEN GO TO
     3230
3220 IF B(X,Y)=0 THEN GO TO
     3260
3230 LET X=X(MF): LET
     Y=Y(MF)
3235 LET B(X,Y)=2: LET
     X=X+DY: LET Y=Y+DX
3240 IF B(X,Y)=2 THEN GO TO
     3260
3250 GO TO 3235
3260 NEXT F
3265 PRINT X(MF),Y(MF): INPUT
     A$
3270 LET CP=1: RETURN
3300 PRINT AT 17,0;"NO
     PUEDO REALIZAR MI
     MOVIMIENTO": FOR F=1
     TO 500: NEXT F
3305 PRINT AT 17,0;"       "
3310 LET CP=1
3320 RETURN
```

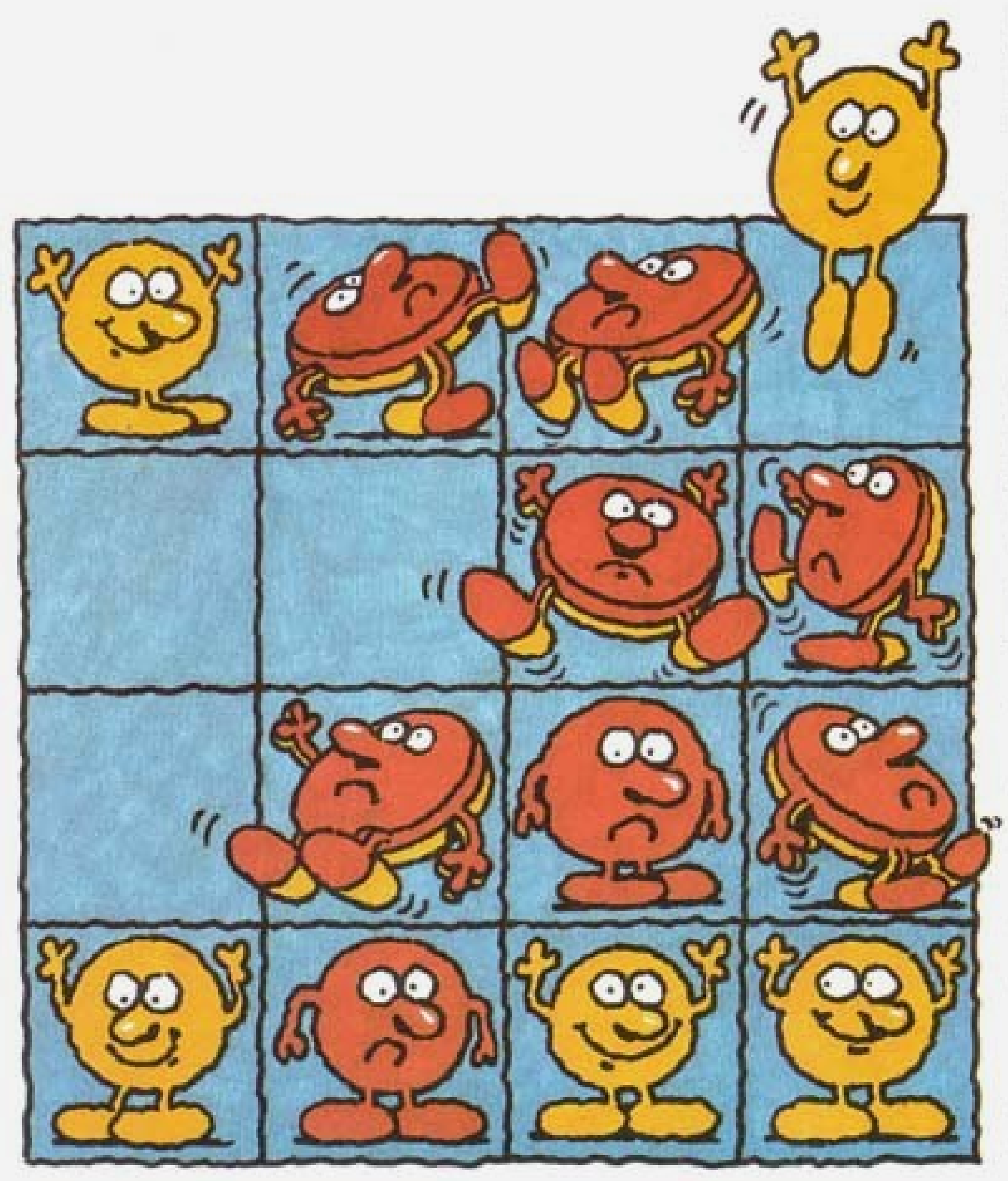
El número de casillas de una fila se coloca a uno, y el número máximo de fichas encontrado en una fila es colocado a cero en la línea 3000. Se ejecutan dos bucles —con las variables de control X e Y—. Éstos recorren el tablero a la búsqueda y captura de espacios libres. Ésta es la sección del programa que más tiempo consume. A medida que el número de casillas vacías disminuye con el avance del juego, también disminuye el tiempo que el ordenador emplea para realizar sus movimientos.

La línea 3010 comprueba si la casilla está vacía. Si no es así, el ordenador salta a la siguiente sentencia NEXT en la línea 3070. Las líneas 3020 a 3060 comprueban si la casilla está al final de una fila que el ordenador pudiera tomar. XP e YP se utilizan igual que antes. DX y DY representan el contenido de la dirección de campo D() y tienen una reserva bastante grande de espacio.

La línea 3030 comprueba si la casilla a verificar se encuentra en el tablero. Si no es así, se comprueba la próxima dirección. Si la casilla comprobada es una casilla perteneciente al jugador, la rutina retrocede entonces a la línea 3030 para comprobar si la siguiente casilla está ocupada por el jugador.

En la línea 3050 se comprueba el tablero para ver si está ocupado por el ordenador. Si es así, y se encuentra una fila, se registran las posiciones de inicio en X() e Y(), y los números de





dirección se graban en N(). También se incrementa el marcador que indica el número de coordenadas encontradas.

Sólo se graba la primera fila, a fin de asegurar que la búsqueda requiera el menor tiempo posible.

Las líneas 3080 a 3110 hallan cuál es el movimiento que da la mayor puntuación en línea recta. Se activa un bucle de uno a NF (número de casillas encontradas). X e Y se equiparan a X(F) y a Y(F). Las coordenadas de dirección, DX y DY, son colocadas en las direcciones indicadas en N(F). CF, a cada ejecución del bucle se coloca a cero un marcador temporal.

La línea 3090 se comprueba para ver si las fichas verificadas pertenecen al jugador. Si es así, se incrementa CF y se comprueba la siguiente casilla en esta dirección. La línea 3100 comprueba si el número encontrado (CF) es mayor que el máximo anterior (MX). En caso afirmativo, MX se coloca igual a CF, y MF igual a las coordenadas de la *mejor* ficha colocada en el índice F del bucle.

La rutina de movimiento se ejecuta en las líneas 3180 a 3260. La línea 3180 ejecuta un bucle de uno a ocho. Ello se realiza de tal modo que se puedan encontrar filas en todas direcciones.

X, Y, DX y DY se colocan como antes, y en la línea 3195 se comprueban X e Y para verificar que aún siguen en el tablero. Si no es así, el programa salta al NEXT en la línea 3260.

La línea 3200 comprueba si el jugador ocupa esta casilla. En caso afirmativo, la rutina salta para intentar la siguiente casilla de esta fila. En este punto ninguna casilla sufre alteración, pues la rutina sólo está probando.

Si la fila acaba con una casilla ocupada por el ordenador, se restan X e Y y las líneas 3225 a 3250 alteran todas las casillas de la fila. Si se encuentra una casilla vacía, se prueba con la siguiente dirección.

Una vez que la rutina ha decidido la casilla a la que desea moverse, la línea 3265 edita las coordenadas y espera a que se pulse la tecla ENTER. La línea 3270 coloca el *flag* CP a uno para el turno del jugador, regresando seguidamente al bucle principal.

EL FINAL DEL JUEGO

```
4000 IF PS>CS THEN GO TO
     5000
4010 IF PS=CS THEN GO TO
     6000
4020 PRINT AT 17,0; INK 2;
     "FUE FACIL!"
4030 PRINT "QUIERES JUGAR
     OTRA VEZ('S'/'N')?"
4040 LET A$=INKEY$: IF
     A$<>"S" AND A$<>"N"
     THEN GO TO 4040
4050 IF A$="S" THEN RUN
4060 STOP
5000 PRINT AT 17,0; INK 2;
     "TUVISTE SUERTE!"
5010 GO TO 4030
6000 PRINT AT 17,0; INK 2;
     "EMPATAMOS, NECESITO
     MAS PRACTICA": GO TO
     4030
```

El final de la rutina del juego se halla a partir de la línea 4000 en adelante. La propia línea 4000 comprueba si el jugador ha ganado comparando PS y CS. El programa salta a la línea 5000 para editar el mensaje de quién es el ganador. La línea 4010 comprueba un dibujo y la rutina del mensaje se encuentra en la línea 6000. Si el ordenador ha vencido, el programa llega a la línea 4020 y edita un mensaje para ¡echar aún más sal en la herida!

Las líneas siguientes son simplemente una opción de ¿otra partida?

MANEJANDO LOS NUMEROS ROMANOS

- REGLAS DE LOS NUMEROS ROMANOS
- MANEJO DEL PROGRAMA
- COMENTARIOS AL LISTADO DEL PROGRAMA

No es tan fácil trabajar con los números romanos. Lo único que hace falta es conocer las reglas que los rigen. Te las vamos a enseñar y además te ofrecemos un programa para que te diviertas con ellos.

Los romanos se distinguieron en muchos campos como pudiera ser el Derecho, la organización militar o las grandes obras públicas, pero en el campo de la Matemática realmente no destacaron. Entre otras razones, quizás una sea su sistema de numeración. Los números romanos son muy complicados de manejar a la hora de hacer operaciones con ellos e incluso resultan muy engorrosos simplemente utilizarlos como tal. Hay que reconocer que los árabes les «mojaron la oreja» sin ningún género de duda. Resulta de una claridad meridiana que 3.933 es mucho más manejable que MMMCMXXXIII y que resulta más intuitivo el manejo de múltiplos de 10, por ejemplo 3.500, 350, 35 que sus equivalentes romanos MMMD, CCCL, XXXV.

A pesar de todo, de lo que no podemos es renegar de uno de los pilares básicos de nuestra civilización y, nos guste o no, lo cierto es que con cierta frecuencia necesitamos leer números romanos (por ejemplo para referirnos a fechas) o incluso expresar un número en cifras romanas. Con ayuda de este programa ya no necesitaremos sudar tinta recordando aquellos que aprendimos en el colegio hace un montón de años o pasar una cierta vergüenza preguntando si es que no encontramos un libro a mano para refrescar nuestros conocimientos.

Las reglas para escribir con números romanos son algo laboriosas, aunque fáciles de entender:

— Los números básicos se expresan mediante las siguientes letras: I, V, X, L, C, D, M que equivalen respectivamente a 1, 5, 10, 50, 100, 500 y 1000.

— Cualquiera de esos símbolos multiplica su valor por 1.000 o por 1.000.000 si se le superpone "—" o "==" respectivamente. No olvidemos que para 1.000, 2.000 y 3.000 existe M, MM y MMM y que estos mismos símbolos con un guión encima equivalen a uno, dos y tres millones.

— Los símbolos «I», «X», «C» y «M» sólo pueden repetirse un máximo de tres veces consecutivas y el resto sólo una.

— Un símbolo a la izquierda de otro le resta su valor a éste (una sola vez).
Solamente se puede anteponer un símbolo cuando no se pueda llegar al mismo resultado comenzando la numeración de izquierda a derecha partiendo de un valor inferior. Ej.: no se puede escribir 999 con «IM» (1.000−1).
Lo correcto es «CMXCIX» (900 +90+9).

— Análogamente al caso anterior, a la derecha se suma pudiendo llegarse, tal como ya se ha dicho, a yuxtaponer hasta tres de algunos símbolos. Ej.: XIII=13, CL=150 (es erróneo «LLL»).

MANEJO DEL PROGRAMA

Después de la carátula de presentación aparece un menú con las siguientes opciones:

1. NÚMEROS ROMANOS A ÁRABES

2. NÚMEROS ÁRABES A ROMANOS

3. TABLA BASE DE CONVERSIÓN

Considerando que éste es un programa fundamentalmente didáctico se ha limitado la numeración de 1 a 3999 para evitar tener que utilizar símbolos con "—" o "=" lo cual, por otra parte no es muy frecuente utilizar en la práctica y además, salvo que se haga notar, puede inducir a error (a veces colocamos los números romanos entre dos rayas horizontales para remarcarlos y no es nuestra intención multiplicar su valor por 1.000 o incluso por 1.000.000).

En cualquier caso ambos límites están protegidos y el ordenador nos solicita un valor adecuado de forma automática.

En la opción 2 el ordenador también nos avisa con el letrero «NÚMERO ERRÓNEO» si pretendemos que nos dé el equivalente a un número fuera de los límites especificados, o lo que es peor, si se trata de una colección de letras que no tienen equivalente o incluso si las hemos colocado de forma que no guardan las reglas del juego.

Mediante la opción 3 podemos visualizar en pantalla las combinaciones básicas árabe/romano.

COMENTARIOS AL PROGRAMA

El listado del programa está suficientemente detallado y documentado como para entenderlo sin ninguna dificultad e incluso para adaptarlo al gusto del lector, pero no obstante vamos a hacer unos comentarios generales sobre la estructura del programa.

La carátula de presentación está temporizada con PAUSE 250, pero podemos evitar el tiempo de espera pulsando ENTER.

La opción elegida en el menú (L240-L315) selecciona el módulo correspondiente.

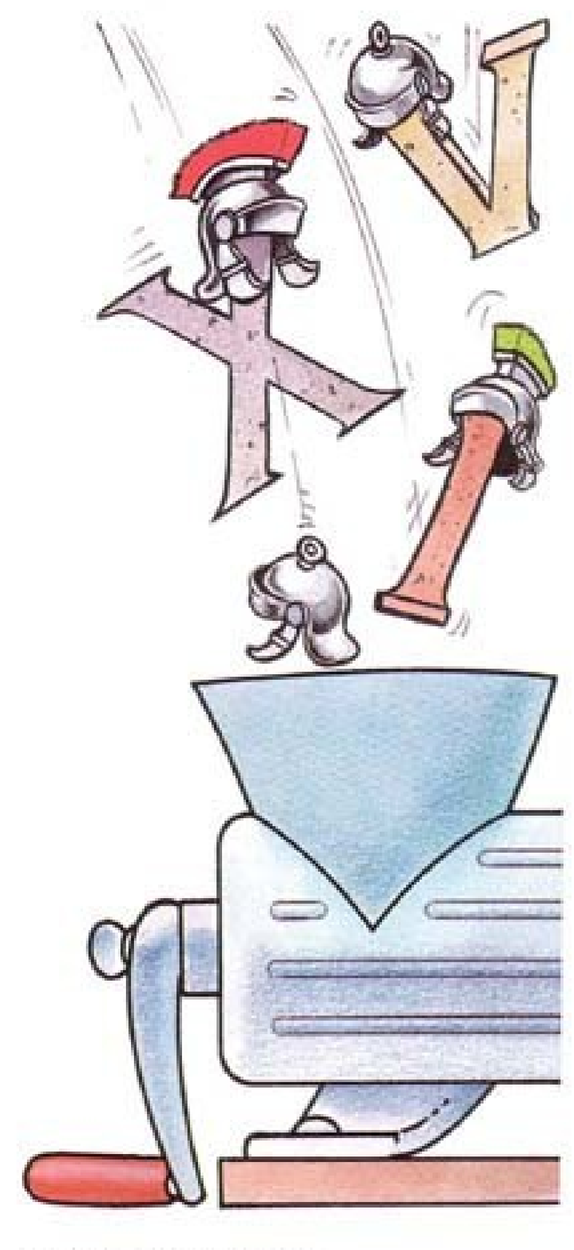
La base de las diversas operaciones se centra en una tabla de conversión compuesta por dos matrices, T(.) y T$(.) donde se almacenan respectivamente los valores árabes y romanos de los valores básicos, visualizados mediante la opción 3.

Para la conversión árabe/romano se separan los millares, centenas, decenas y unidades, y se almacenan para su posterior manipulación en la matriz C(.). La separación se consigue mediante la línea 520, situando en C(1) la cifra correspondiente a los millares, en C(2) las centenas, hasta llegar a C(4) con las unidades.

Luego se hace la búsqueda respectiva en la matriz T(.), es decir, se va comparando cada elemento de C(.) con los 30 valores de T(.) hasta encontrar el adecuado, momento en el que se toma el valor correspondiente en T$(.) y se yuxtapone a S$, que es la string que va almacenando nuestro número romano. Como puedes observar en la línea 560, no se toma toda la longitud del valor correspondiente de T$(.), ya que esta matriz se ha definido de longitud fija 4 y no siempre estará llena del todo. Así, antes llamamos a la subrutina ubicada en la línea 1000 la cual nos dará la longitud de la string K$ (con el valor de T$(.) a unir). No es lo mismo sumar a S$ el valor de T$(29)="MM " con dos espacios, que el de K$(1 TO 2)="MM".

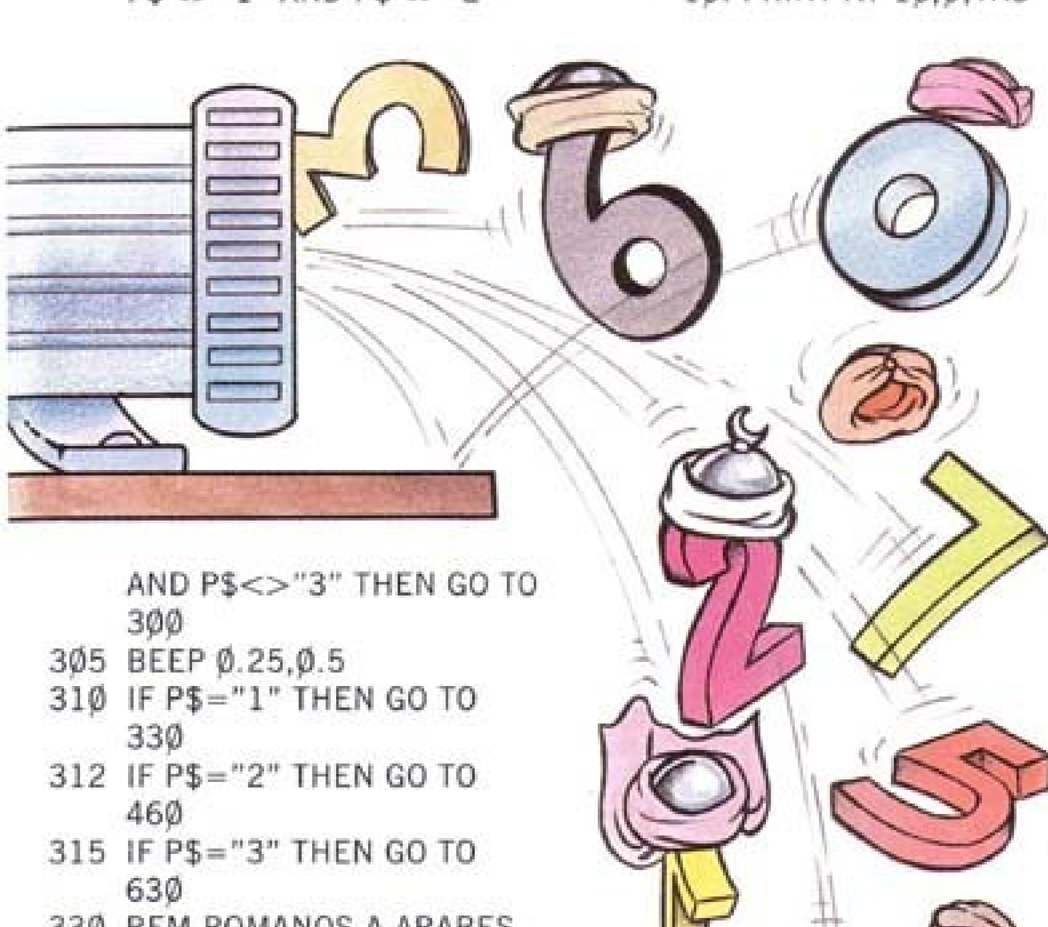
El proceso es análogo para la conversión romano/árabe, aunque es más laborioso. Se comienza explorando la matriz T$(.) por su valor más elevado (MMM) y se analiza si dicha combinación se halla en el número romano dado, y precisamente en su parte izquierda más extrema (Bucle 370 a 410). Si esto ocurre tomamos el valor numérico correspondiente de T(.), lo sumamos a N y eliminamos las letras que ya hemos valorado (línea 390) quedándonos con la parte derecha de la matriz, quitando ese K$(1 TO LK) (la misma operación que en la conversión árabe/romano, subrutina 1000). Continuamos así hasta el final de la tabla T$(I), incrementando el valor numérico que vamos obteniendo. Hemos de tener en cuenta que si hemos encontrado una cifra válida para los millares, no pasaremos a comprobar el siguiente millar, sino la primera cifra de las centenas, y lo mismo con las decenas, unidades... No sería válido el número romano «CCCCC» por lo que una vez encontrado «CCC» en T$(.) sumaremos 300 a N, pero no podremos seguir comprobando la cifra de las centenas «CC» que sería 200, sino que pasaríamos a comprobar si existe en S$ la más alta cifra de las centenas de T$(.) que sería «XC», 90 en árabe. El control de esta operación lo gestiona la subrutina situada en la línea 3000. Si al final del proceso no quedan letras por valorar en el número romano, se presenta el resultado en pantalla. En caso contrario el número que hemos calculado hasta ese momento no es bueno y por lo tanto quiere decir que el conjunto de letras que pretendían ser un número romano válido tampoco lo son.

```
  5 REM  ==============
 1Ø REM  PROGRAMA DE
         CONVERSION DE
         NUMERACION ARABE Y
         VICEVERSA
 2Ø REM  ==============
 3Ø REM INPUT SINCLAIR /
    MOLISOFT FEBRERO 87
 4Ø REM  ------------------------
 5Ø PAPER 1: INK 9: BORDER 1:
    CLS : BEEP Ø.5,1Ø
 6Ø REM PRESENTACION
 7Ø PRINT AT 5,Ø; "************
    ********************"
 8Ø FOR I=1 TO 14: PRINT "*";
    TAB 31;"*": NEXT I
 9Ø PRINT AT 2Ø,Ø; "********
    ************************"
    1ØØ PRINT AT 8,4; PAPER
     6;"CONVERSION DE
     NUMERACION"
11Ø PRINT AT 12,4; INVERSE 1;
    "ARABE => ROMANA =>
    ARABE"
12Ø PRINT AT 15,9; PAPER 4;
    "INPUT SINCLAIR"
13Ø PRINT AT 2Ø,18; INVERSE
    1;"Molisoft 1987"
14Ø PAUSE 25Ø: CLS : REM
    TIEMPO DE ESPERA
15Ø DIM T(3Ø): DIM T$(3Ø,4):
    DIM C(4)
```



```
16Ø REM DATOS DE
    EQUIVALENCIAS
17Ø DATA  1,"I",2,"II",3,"III",
    4,"IV",5,"V",6,"VI",7,
    "VII",8,"VIII",9,"IX",1Ø,"X"
18Ø DATA  2Ø,"XX",3Ø,"XXX",
    4Ø,"XL",5Ø,"L",6Ø,"LX",
    7Ø,"LXX",8Ø,"LXXX",9Ø,
    "XC",1ØØ,"C"
19Ø DATA  2ØØ,"CC",3ØØ,
    "CCC",4ØØ,"CD",5ØØ,"D",
    6ØØ,"DC",7ØØ,"DCC",8ØØ,
    "DCCC",9ØØ,"CM",1ØØØ,
    "M"
2ØØ DATA  2ØØØ,"MM",3ØØØ,
    "MMM"
```

```
210 FOR I=1 TO 30
220 READ T(I),T$(I)
230 NEXT I
240 REM MENU
250 CLS : PRINT AT 5,8; PAPER
    6;"CAMBIO DE
    NUMERACION"; PAPER 1;
    TAB 8;"--------------------"
260 PRINT AT 8,3;"1 - ";
    INVERSE 1;"NUMEROS
    ROMANOS A ARABES"
270 PRINT AT 10,3;"2 - ";
    INVERSE 1;"NUMEROS
    ARABES A ROMANOS"
280 PRINT AT 12,3;"3 - ";
    INVERSE 1;"TABLA BASE
    DE CONVERSION"
290 PRINT AT 21,0; PAPER 7;
    INK 2;"PULSE LA OPCION
    DESEADA"
300 LET P$=INKEY$: IF
    P$<>"1" AND P$<>"2"
    AND P$<>"3" THEN GO TO
    300
305 BEEP 0.25,0.5
310 IF P$="1" THEN GO TO
    330
312 IF P$="2" THEN GO TO
    460
315 IF P$="3" THEN GO TO
    630
330 REM ROMANOS A ARABES
340 CLS : PRINT AT 0,2; PAPER
    6;"NUMERO ROMANO";
    PAPER 1;"   "; PAPER 6;
    "NUMERO ARABE"
350 PRINT TAB 2;
    "=============
    ============":
    PRINT
360 INPUT PAPER 4; INK 0;
    "NUMERO ROMANO?
    (P=MENU) ";N$: IF
    N$="P" THEN GO TO 250
365 IF N$="" THEN GO TO 360
370 LET S$=N$: LET N=0
380 FOR I=30 TO 1 STEP -1:
    GO SUB 1000
385 IF LK>LEN S$ THEN GO TO
    410
390 IF S$(1 TO LK)=K$(1 TO
    LK) THEN LET N=N+T(I):
    LET S$=S$((LK+1) TO ):
    GO SUB 3000
400 IF S$="" THEN LET I=1
410 NEXT I
420 PRINT AT 10,0;"       ": REM
    32 ESPACIOS EN BLANCO
430 IF S$<>"" THEN BEEP 1,
    -10: PRINT AT 10,0;TAB
    (15-LEN N$); PAPER 7;
    N$; PAPER 1;TAB 18;
    PAPER 7; INK 0;"NUMERO
    ERRONEO": GO TO 360
440 BEEP 0.25,0.5: PRINT AT
    10,0;TAB (15-LEN N$);
    PAPER 7;N$; PAPER 1; TAB
    18; PAPER 7;N
450 GO TO 360
460 REM ARABES A ROMANOS
470 CLS : PRINT AT 0,2; PAPER
    6;"NUMERO ARABE";
    PAPER 1;"   "; PAPER 6;
    "NUMERO ROMANO"
480 PRINT TAB 2;
    "============
    =============":
    PRINT
490 INPUT PAPER 4; INK 0;
    "NUMERO (0<N<3999)
    0=MENU ";N
495 LET N=INT N: IF N=0
    THEN GO TO 250
500 IF N<1 OR N>=4000
    THEN GO TO 490
510 LET N$="": LET H$=STR$
    N: LET H=LEN H$
520 LET R=N: LET
    C(1)=1000*INT (R/1000):
    LET R=R-C(1): LET
    C(2)=100*INT (R/100): LET
    R=R-C(2): LET
    C(3)=10*INT (R/10): LET
    C(4)=R-C(3)
530 FOR L=1 TO 4
540 FOR I=1 TO 30
550 LET S$=""
560 IF T(I)=C(L) THEN GO SUB
    1000: LET S$=K$(1 TO
    LK): LET I=30
570 NEXT I
580 LET N$=N$+S$
590 NEXT L
600 PRINT AT 10,0;"   ": REM
    32 ESPACIOS EN BLANCO
610 BEEP 0.25,0.5: PRINT AT
    10,(14-H); PAPER 7;N;
    PAPER 1;TAB 17; PAPER 7;
    N$
620 GO TO 490
630 REM TABLA BASICA DE
    CONVERSION
```



```
640 CLS : PRINT TAB 4; PAPER
    6;"TABLA BASICA DE
    CONVERSION": PRINT
650 PRINT TAB 3; PAPER 4;
    "NUMERO ARABE"; PAPER
    1;"    "; PAPER 4;"NUMERO
    ROMANO"
660 PRINT TAB 3;"=======
    =============
    =====": PRINT
670 FOR I=1 TO 3Ø
675 GO SUB 1ØØØ: GO SUB
    2ØØØ
680 PRINT TAB (1Ø-V); PAPER
    7;T(I); PAPER 1;TAB 23;
    PAPER 7;K$(1 TO LK)
690 PAUSE 25: BEEP Ø.25,Ø.5
700 NEXT I: PRINT : PRINT
710 PRINT AT 21,Ø; PAPER 2;
    INK 7; BRIGHT 1; FLASH 1;
    "PULSE UNA TECLA"
720 PAUSE Ø: GO TO 25Ø
730 GO TO 25Ø
74Ø STOP
1ØØØ LET K$=T$(I): LET LK=Ø
1Ø1Ø FOR Q=1 TO 4: IF
     K$(Q)<>" " THEN LET
     LK=LK+1: NEXT Q
1Ø2Ø RETURN
2ØØØ LET V$=STR$ T(I): LET
     V=LEN V$: RETURN
3ØØØ IF I>28 THEN LET I=27
3Ø1Ø IF I>19 AND I<28 THEN
     LET I=18
3Ø2Ø IF I>1Ø AND I<19 THEN
     LET I=9
3Ø3Ø IF I>1 AND I<1Ø THEN
     LET I=1
3Ø4Ø RETURN
```

NO OLVIDES EL TELEFONO... 

Cuando, por cualquier motivo, nos escribas, no olvides indicar tu número de teléfono. Así nos será más fácil y rápido ponernos en contacto contigo. Gracias.

GANADORES DE LOS MEJORES DE INPUT SINCLAIR

En el sorteo correspondiente al número 15 entre quienes escribisteis mandando vuestros votos a LOS MEJORES DE INPUT han resultado ganadores:

NOMBRE	LOCALIDAD	JUEGO ELEGIDO
Julio Soto Vidal	Cartagena (Murcia)	FIRE LORD
Albert Capilla López	Sant Carles de la Rápita (Barna)	JAIL BREAK
Roger Fernández Espuny	Sant Boi de Llobregat (Barna)	TWO ON TWO
Josep Fité Aulina	Olot (Girona)	LEADER BOARD GOLF
Marc Grau Bel	Sant Boi de Llobregat (Barna)	GREEN BERET
Luis Miguel Díaz Blanco	Madrid	GAME OVER
J. Carlos Pérez Álvarez	O Barco de Veras (Orense)	COBRA
José Luis Ares Martín	Alcorcón (Madrid)	BASKETBALL
David Asensio Reyes	Cádiz	GHOST & GOBLINS
Jordi Muñoz Arcarons	Masnou (Barcelona)	THE GOONIES

LOS MEJORES DE INPUT SINCLAIR

PUESTO	TÍTULO	PORCENTAJE
1°	COMMANDO	26,9 %
2°	GREEN BERET	13,5 %
3°	SABOTEUR	11,6 %
4°	GHOSTS'N GOBLINS	8,9 %
5°	SIR FRED	8,3 %
6°	RAMBO	7,7 %
7°	AVENGER	7 %
8°	COBRA	5,8 %
9°	SKY FOX	5,8 %
10°	PHANTOMAS II	4,5 %
		100,0 %

Para la confección de esta relación únicamente se han tenido en cuenta las votaciones enviadas por nuestros lectores de acuerdo con la sección «LOS MEJORES INPUT».

Marzo de 1987



LA MUSICA, LOS MICROS Y EL MIDI

El hacer música con tu ordenador no es más que un punto de partida. La introducción de equipos basados en el MIDI te permite conectar los ordenadores con los sintetizadores musicales y pasar de uno a otro.

El sonido se ha convertido en una de las principales características que la mayoría de la gente espera encontrar en un microordenador doméstico, hasta el punto de que hay personas que pueden sentirse inclinadas a comprar un ordenador determinado a causa de sus posibilidades sonoras. Aparte de la música que puedas hacer con tu ordenador, está empezando a introducirse y a estar cada vez más fácilmente disponible la posibilidad de conectar muchos microordenadores domésticos a sintetizadores y a otras clases de instrumentos musicales. El estándar correspondiente se llama MIDI, Interface Digital para Instrumentos Musicales (Musical Instrument Digital Interface) y te brinda una nueva gama de posibles utilizaciones para tu ordenador.

EL SONIDO DE LOS ORDENADORES

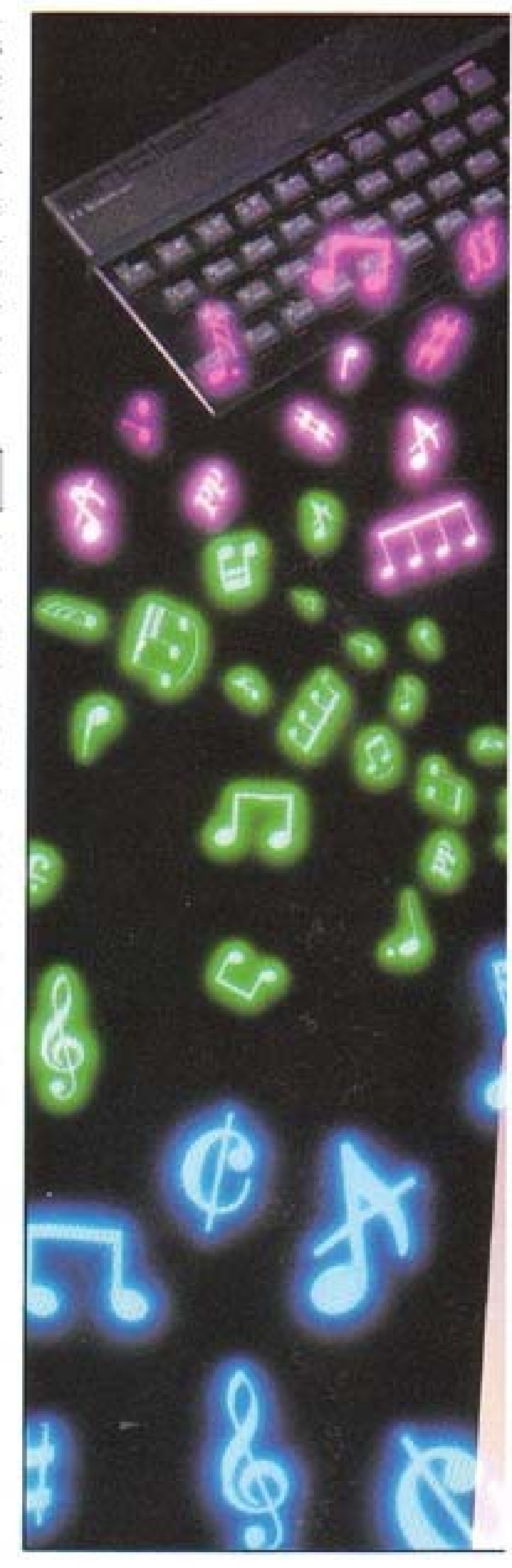
El sonido que son capaces de producir los ordenadores ha evolucionado enormemente desde los pitidos y zumbidos producidos por los primeros modelos que incorporaban sonidos, hasta la posibilidad de ejecutar trozos musicales y producir otros efectos sonoros. El Spectrum te ofrece unos sonidos muy sencillos por medio del comando BEEP y sus notas siempre suenan iguales; se trata de un auténtico tono electrónicamente puro. Además en este ordenador no se puede producir desde el BASIC más que una sola nota al mismo tiempo.

Si has probado alguno de los programas de música que hemos publicado en INPUT, ya sabrás lo que es posible conseguir con tu ordenador. Incluso los sonidos más sofisticados producidos con el más sofisticado de los ordenadores, son de una calidad inferior a la que nos tienen acostumbrados las grabaciones normales, por no mencionar los inconvenientes de intentar ejecutar un fragmento musical con un teclado QWERTY. Incluso en el caso más favorable, el micro sigue estando por debajo de un instrumento musical construido deliberadamente para ser utilizado como tal.

INSTRUMENTOS MUSICALES

La historia del desarrollo de los instrumentos musicales en las últimas décadas ha seguido un desarrollo bastante paralelo al de las máquinas de calcular. Los instrumentos tradicionales eran mecánicos: trozos de piel que se golpeaban, cuerdas pulsadas o frotadas, etc. Poco a poco, la necesidad creciente de conseguir un mayor volumen en los conciertos en vivo y la necesidad de hacer grabaciones, ha ido conduciendo a que se vayan electrificando instrumentos como guitarras y pianos, hasta disponer finalmente, en los últimos años, de instrumentos musicales puramente electrónicos tales como los sintetizadores. De la misma forma que las máquinas de calcular han evolucionado desde el ábaco mecánico hasta el moderno ordenador que incorpora tecnología electrónica digital, los últimos instrumentos musicales que han aparecido están llenos de circuitos integrados.

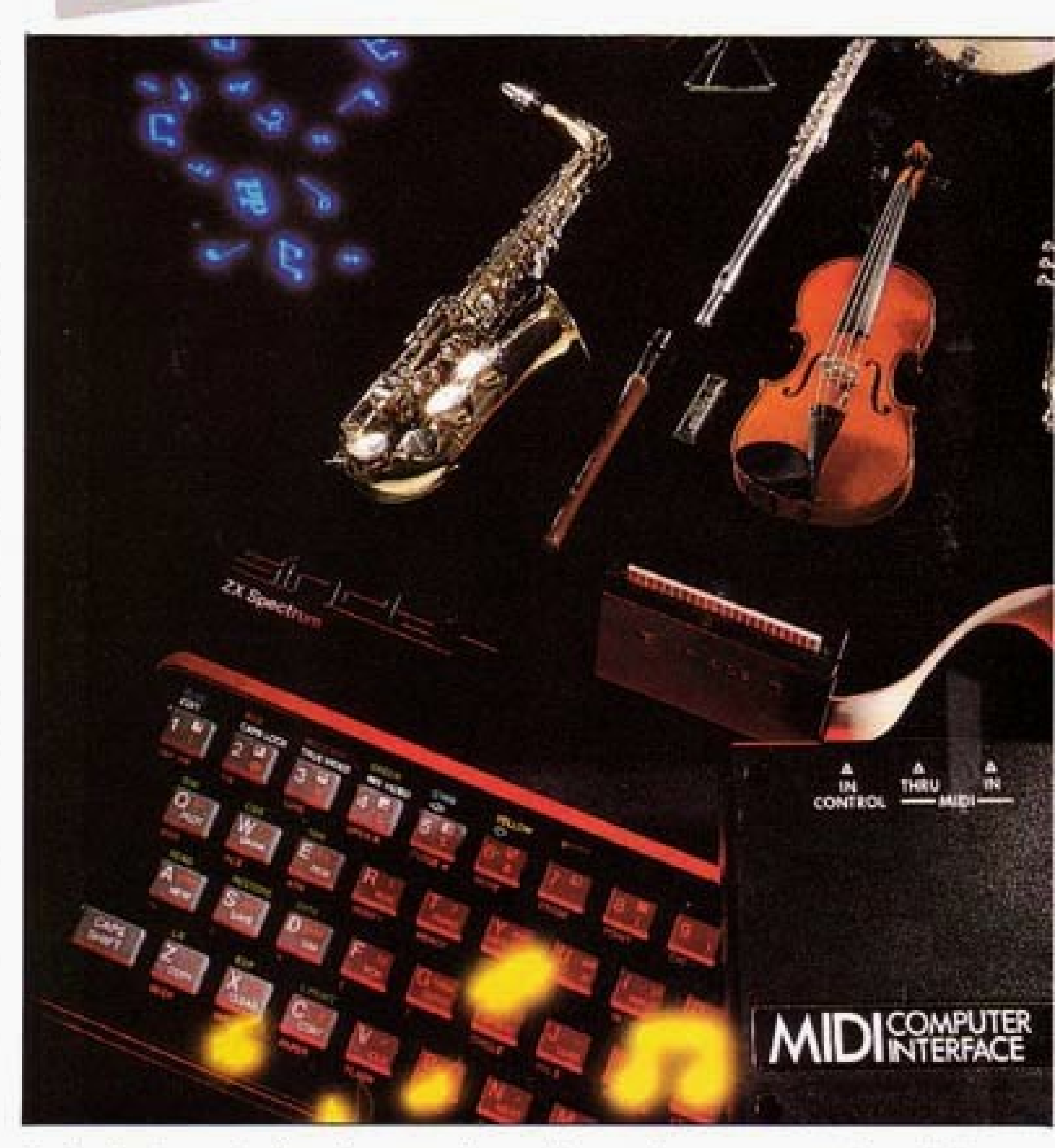
Los sintetizadores modernos son unos dispositivos extraordinariamente sofisticados. En lugar del número limitado de notas que pueden ser interpretadas en un ordenador, y del número limitado (o inexistente) de tipos de envolventes, te encontrarás con un asombroso conjunto de posibilidades.





MIDI COMPUTER INTERFACE
Roland
RHYTHM COMPOSER TR-707

Un sintetizador típico de precio medio te permitirá la ejecución de acordes de hasta ocho notas en un teclado adecuado. Casi todas las máquinas ofrecen un conjunto de sonidos preprogramados, por lo que si quieres tener el sonido de un piano o un violín no tienes más que pulsar la tecla adecuada. Pero no todo se queda en los sonidos preprogramados; puedes jugar con las ideas que se te ocurran hasta producir casi cualquier sonido que desees. El sueño del publicista con un sintetizador que puede llegar a ser una orquesta completa, no está disponible todavía, pero probablemente está ya acechando en la mente de alguien en alguna parte.



Cuando se mencionan los sintetizadores, casi todo el mundo piensa inmediatamente en los instrumentos de teclado que constituyen con mucha diferencia el tipo más corriente de sintetizador. Pero dado que realmente el corazón del sintetizador es una caja que contiene una electrónica para producir sonidos, la cual puede ser atacada por algún tipo de señal, en pura teoría no hay razón para que no se utilice otro tipo de instrumentos para atacar dicha electrónica. En la práctica, la cosa es ligeramente diferente. Por diversas razones técnicas, el tipo de sintetizador musical más popular sigue siendo el de teclado, si bien puedes comprar sintetizadores tipo guitarra, que se tocan exactamente igual que una guitarra aunque suenan como tú quieras que suenen. También hay sintetizadores de percusión, diferentes de las máquinas de percusión sintética, que se tocan golpeando sobre una serie de almohadillas.

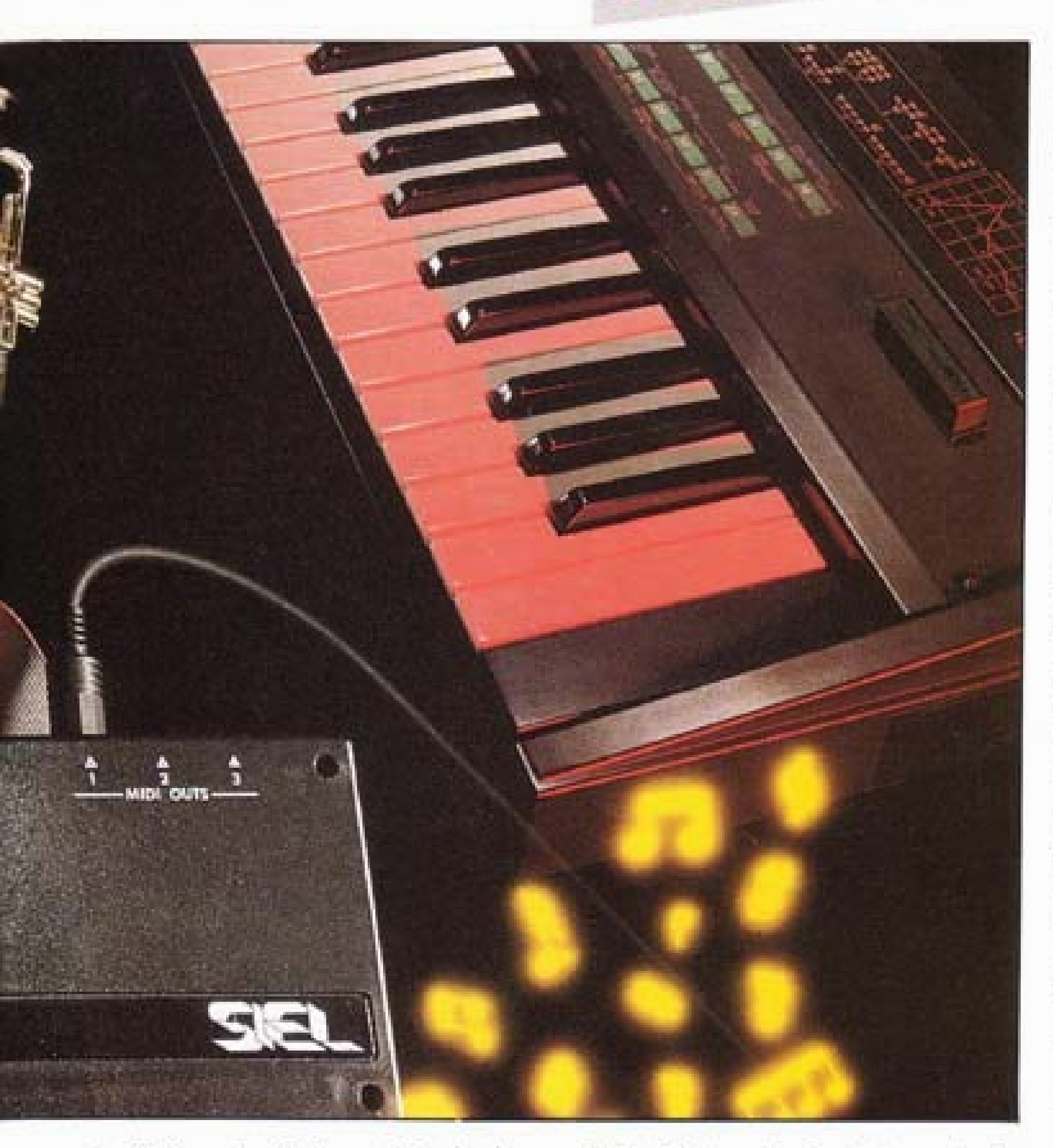
Por otra parte las máquinas de percusión están preprogramadas para proporcionar un soporte rítmico sin la intervención de un ejecutante. Pero también éstas caen dentro de la categoría de sintetizadores. Hasta hace poco tiempo, las máquinas de percusión tenían un sonido muy característico, por lo que cualquier grabación hecha con una máquina de éstas en vez de con un verdadero instrumento de percusión era reconocible al instante. Con los últimos avances de la tecnología, ya no puede decirse esto. Casi todas las máquinas de percusión ofrecen unos cuantos ritmos preseleccionados, poseyendo además una capacidad de memoria que te permite crear y almacenar tus propios esquemas rítmicos.

Esta última posibilidad de las máquinas de percusión señala el camino hacia una verdadera revolución en lo musical. Hasta ahora la habilidad musical siempre había dependido de algún tipo de destreza manual, es decir de la posibilidad de mover los dedos con rapidez y precisión sobre un teclado, o de poder golpear con precisión una membrana en el instante justo. En los instrumentos de viento, la habilidad consiste en coordinar el movimiento de los dedos con los diferentes movimientos respiratorios. La llegada de los instrumentos musicales programables modifica por completo este panorama.

La máquina programable de percusión no sirve como sustituto del talento musical. Sigue haciendo falta alguien capaz de entender el ritmo y de «oír» en su mente el efecto deseado. Lo que hace la máquina es liberar a la persona dotada de talento musical de esa dependencia de la habilidad manual y de la necesidad de adquirir un instrumento de percusión caro y voluminoso.

Los sintetizadores de teclado tienen ya la posibilidad de sonar virtualmente como cualquier instrumento que tú elijas. Si a ello se le añade la posibilidad de programar sus ejecuciones, se abre todo un nuevo mundo musical incluso ante los que parecen tener una mano con cinco dedos gordos. Entra a MIDI, un sistema que te permite incorporar la programación en el sintetizador.

Ahora que los ordenadores y los instrumentos musicales emplean el mismo tipo de tecnología, es relativa-



mente fácil enviar información desde un ordenador a un instrumento musical y viceversa; ésta es la función de MIDI.

¿QUE ES EL MIDI?

El MIDI es un estándar de interface, como ocurre con la Centronics o la RS 232 con las que ya te habrás encontrado en relación con periféricos tales como impresoras y módems. En este caso, aunque el MIDI se utiliza exclusivamente en conexión con el mundo de la música, el papel de la interface es exactamente el mismo: la transferencia de información de un sitio a otro, con un formato normalizado. Su principal función como estándar es garantizar que la información se transmita de tal forma que cualquier tipo de equipo compatible-MIDI sea capaz de entender la información que recibe.

Ha habido varios intentos anteriores para imponer una norma de comunicación entre instrumentos musicales, pero ninguno de ellos ha conseguido lograr una aceptación generalizada. En el caso del MIDI, las perspectivas parecen ser diferentes, ya que todos los sintetizadores y máquinas de percusión producidas por los dos principales fabricantes de teclados del mundo, Yamaha y Roland, se adaptan a la nueva norma y actualmente está haciendo su aparición una nueva gama de ordenadores compatibles con dicha norma. Parece ser que la norma es universalmente aceptada, tanto por los fabricantes de instrumentos musicales electrónicos como por los de ordenadores susceptibles de ser conectados a ellos.

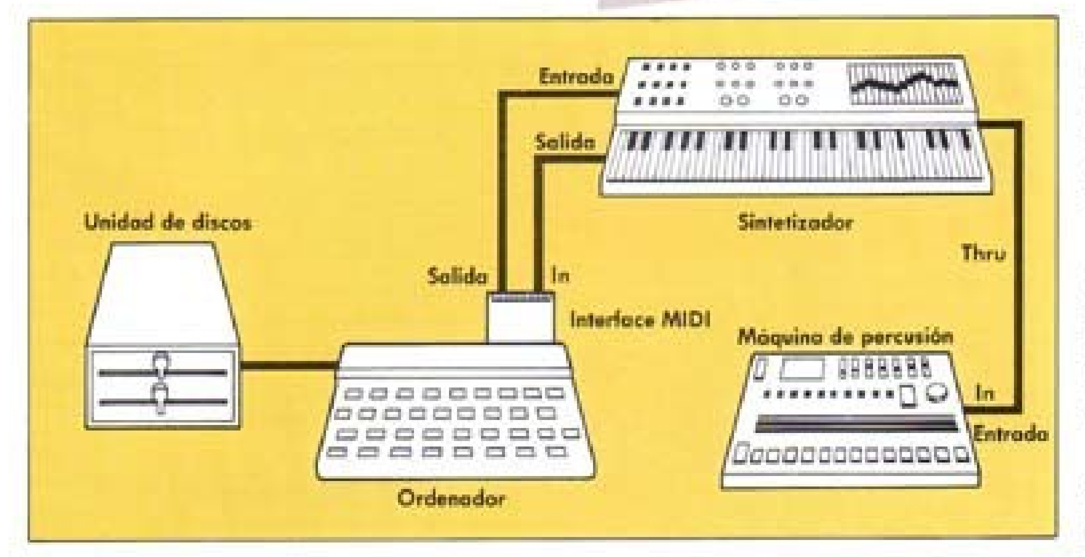
Todo equipo compatible con MIDI tiene tres conectores DIN de cinco pines, que llevan las denominaciones «IN», «OUT» y «THRU» (algunos equipos MIDI más antiguos puede que no lleven el conector «THRU»). El conector «IN» permite que el equipo reciba señales MIDI desde otro equipo MIDI. Con «OUT» ocurre exactamente lo contrario, permitiendo que un equipo MIDI envíe señales MIDI a otro equipo MIDI. Por medio de «THRU» se envía una copia directa de la información que le llega a un equipo MIDI hacia otro equipo MIDI. Esto significa que se puede actuar sobre varios dispositivos al mismo tiempo, conectándolos a través de sus conectores «THRU». Por eso los equipos que no dispongan del conector «THRU» son mucho más limitados en sus prestaciones.

El estándar MIDI permite transmitir separadamente hasta 16 canales de información al mismo tiempo. Cada canal permite al músico controlar un instrumento por separado, pero la información coexiste sobre el mismo hilo. Cada dispositivo «se sintoniza» con la información que se le está enviando, de una forma algo semejante a la que tiene un televisor de sintonizarse en un determinado canal.

¿COMO SE UTILIZA EL MIDI?

El MIDI está en el mercado desde 1982, aunque en principio sólo llamó la atención de unos cuantos propietarios de micros domésticos. Los músicos han utilizado el MIDI para atacar unos instrumentos a partir de otros. Por ejemplo, se puede hacer que dos sintetizadores produzcan sonidos diferentes, pero al mismo tiempo a partir de un solo teclado, conectando para ello ambos instrumentos por medio de una interface MIDI y tocando en el teclado de uno de ellos.

El MIDI también permite a un músico conectar una máquina de percusión a un teclado y sincronizar un ritmo con una determinada melodía. También es posible conectar un secuenciador. Un secuenciador es un dispositivo capaz de recordar lo que se ha tocado y de volver a ejecutarlo. Los hay de dos tipos: de tiempo real y de pasos. Un secuenciador de tiempo real ejecuta exactamente lo que el músico ha tocado, mientras que un secuencia-



Ejemplo de MIDI enlazando un ordenador, una máquina de percusión y un sintetizador. La información musical se puede almacenar en el disco.

dor por pasos literalmente va recorriendo la melodía paso a paso, mientras el músico va tocando las notas una a una, cubriendo los intervalos individuales de tiempo hasta que se completa toda la melodía.

EL MIDI Y LOS ORDENADORES DOMESTICOS

Coincidiendo con la aparición de los ordenadores que utilizan el estándar MSX se ha dado bastante publicidad a la relación entre el MIDI y los micros domésticos. Yamaha ha introducido el ordenador musical CX5M, que es un ordenador MSX con un sintetizador incorporado. Si a este ordenador se le añade un teclado de tipo pianístico, el propietario se encuentra en posesión de un auténtico sintetizador hecho y derecho. Esta máquina brinda toda clase de posibilidades a los músicos: se puede componer sobre la pantalla de un monitor, o se puede utilizar el ordenador como secuenciador sin necesidad de ningún hardware adicional.

El costo de este ordenador supera considerablemente al de un Spectrum, pero puedes tener un dispositivo parecido si utilizas tu ordenador conectándolo a un sintetizador MIDI. Para ello necesitas una caja de interface MIDI a la que conectar tu ordenador, un conector y un poco de software.

La caja de interface, cuyo coste es inferior al de un Spectrum, te permitirá conectar tu ordenador a cualquier equipo que sea compatible con la interface MIDI. En la actualidad el precio de los sintetizadores compatibles con MIDI es bastante superior al de un Spectrum, por lo que en realidad constituye una vía muy cara para que los usuarios de ordenadores domésticos extiendan el campo de sus habilidades musicales, pero al igual que ocurre con las impresoras, los monitores de color y las unidades de disco, se puede predecir una caída en el precio de los instrumentos. En un futuro no lejano, el precio de un sintetizador resultará probablemente comparable al de los micros domésticos que pueden utilizarse para controlarlo.

Pero incluso antes de que se produzca esta caída de precios, los poseedores de equipos musicales compatibles con MIDI considerarán que una combinación de un micro doméstico y una interface MIDI es una alternativa muy atractiva. Con un software adecuado se puede disponer de un amplio margen de posibilidades y se puede disponer de una gran cantidad de efectos añadidos a un coste muy moderado.

Cuando un ordenador doméstico se conecta a la interface MIDI, no se utilizan para nada las capacidades sonoras que incorpora; el sonido se genera siempre en el sintetizador o en la máquina de percusión, por lo que no es preciso adquirir un ordenador muy caro para utilizarlo específicamente con el MIDI. Es importante darse cuenta de que no hay ninguna ventaja real en utilizar ordenadores profesionales de alto precio en lugar de micros domésticos. Incluso la cantidad de memoria adicional de que disponen los ordenadores profesionales es en gran parte superflua, ya que el espacio de memoria normal de un micro doméstico supera ampliamente, por ejemplo, al de cualquier secuenciador. La idea básica es que un ordenador al que se pueda conectar una interface MIDI es tan bueno como cualquier otro, aunque pudiera ser que en el caso del Spectrum de 16K te encuentres con la capacidad de memoria ligeramente limitada.

Hay que destacar además que la calidad del sonido disponible no está li-

MIDI in
MIDI thru
(CH-1) = canal 1, etc.
MIDI in
MIDI thru
(CH-2)
MIDI in
MIDI thru
(CH-3) C
Salida MIDI
MIDI in
(CH-4) D
Sintetizador

Un bus de MIDI puede contener hasta 16 canales de información que controlen otros tantos instrumentos.

mitada por el medio de grabación. Como el sonido se almacena digitalmente, se le puede comparar con el de un medio como el del Compact Disk, en vez de la cinta a la que se le puede añadir todo tipo de ruido indeseable. En otras palabras, lo que sale de un sistema MIDI es exactamente igual a lo que entra.

SOFTWARE PARA EL «MIDI»

Una vez que ya tengas enganchado tu ordenador a un instrumento musical por medio del MIDI, necesitarás algún software para que funcione. Actualmente la oferta es todavía bastante restringida y comparable en precio al software comercial disponible para ordenadores domésticos. La situación irá cambiando a medida que haya más gente que use el MIDI.

No obstante, incluso con la escasa oferta que existe actualmente, ya hay programas que te permiten duplicar secuenciadores, componer música en varias pistas y editar tus propias melodías. Las ofertas concretas varían de unos fabricantes a otros y de unos ordenadores a otros.

Aunque te consideres incapaz de ejecutar una nota en un instrumento musical, te encontrarás con que puedes interpretar música compuesta sobre el monitor de tu ordenador y enviando la información al instrumento musical que ha de ejecutarla. También puedes almacenar la composición en disco o cinta para hacer después playback o para ejecutarla en otro momento. Se prevé la existencia de hojas de música disponibles en EPROMs codificadas en formato MIDI, por lo que se tendrán piezas completas ejecutables de forma parecida a un disco o cinta, o bien para tocar al mismo tiempo que se hace un playback con el contenido de una EPROM.

Un paquete de software típico del MIDI es el programa compositor de música que te permite ir construyendo tu pieza sobre la pantalla, de forma muy parecida a como la escribirías sobre el papel, incorporando las notas sobre un pentagrama. Unas utilidades completas de edición y ejecución te permiten comprobar el estado de avance e introducir modificaciones en la pantalla.

Pero no termina aquí la cosa, ya que un buen software pondrá además a tu disposición todas las posibilidades del sintetizador. Puedes controlar tantas voces simultáneas como permita el sintetizador; un sintetizador polifónico típico de alta calidad puede llegar a ejecutar hasta 16 notas al mismo tiempo. Puedes escogerlas dentro del margen de notas preprogramadas o combinar nuevas notas. Si tu sintetizador tiene la posibilidad de desdoblar en dos el teclado, puedes tener dos instrumentos diferentes sonando al mismo tiempo, por ejemplo, una melodía y su acompañamiento. En general hay tres tipos de información que puedes enviar al MIDI: notas, cambios de programa y mezclas de sonidos.

En la actualidad hay un conjunto normalizado de códigos MIDI, que funcionarán con cualquier sintetizador compatible-MIDI. Pero dichos códigos sólo sirven para controlar las funciones más básicas disponibles. Se puede acceder a otras prestaciones especiales por medio de sistemas de codificación extendidos, que normalmente varían de un instrumento a otro. Como consecuencia, una pieza compleja con una orquestación variada puede exigir la familiaridad con un gran número de códigos MIDI, si bien es muy posible que esto se simplifique considerablemente en el futuro.

Si te consideras un programador de ordenadores y no un músico, no por ello tienes que desanimarte para utilizar el MIDI. En cierto modo tienes alguna ventaja. Si puedes programar en código máquina, no hay nada que te impida escribir tu propio software para el MIDI, cortado a la medida de tus necesidades, con lo que además te ahorrarás el gasto que supone la compra de un software comercial.

El MIDI parece ofrecer muchas posibilidades al músico como al no músico. A medida que los precios van bajando, los instrumentos musicales parecen estar destinados a abrirse camino en la mayoría de los hogares; ¡quién sabe si llegarán a generalizarse las veladas musicales en torno al sintetizador! Y además tendrás la oportunidad de llegar a ser realmente creativo con tu ordenador.

PROGRAMA ENSAMBLADOR PARA EL SPECTRUM

Si te aburre manejar datos en hexadecimal, ¿por qué no hacer que tu *Spectrum* se encargue de traducir tus listados en lenguaje ensamblador a código máquina, en lugar de tener que hacerlo tú? Incluso calculará la longitud de los saltos.

El ensamblado a mano es una tarea muy tediosa. Incluso aunque te sepas de memoria todos los códigos de operación y estés familiarizado con los diferentes modos de direccionamiento, la traducción a código máquina del listado de un programa largo, escrito en lenguaje ensamblador, y su posterior introducción en el ordenador con ayuda del monitor de código máquina, es un trabajo extraordinariamente laborioso.

Los ordenadores son muy buenos y exactos en la realización de tareas repetitivas, por lo que pueden suplirte en la realización de las mismas. Además puedes utilizar el mismo programa para POKEar al mismo tiempo en la memoria el código máquina resultante.

El programa que sigue a continuación es un ensamblador para el *Spectrum* de 48K. En un *Spectrum* de 16K o en un *ZX81* no hay suficiente espacio en la memoria para ejecutar un programa de estas características. Para que estos programas sean adecuados para su publicación, tienen que estar escritos en BASIC, por lo que no son tan rápidos como los monitores en código máquina comercialmente disponibles.

Sin embargo, funcionan bien y serán capaces de ensamblar los programas escritos en lenguaje ensamblador que te encuentres en los libros o en INPUT. Por lo demás, harías bien en ir a tomarte un café mientras el ensamblador del *Spectrum* se enfrenta con un programa muy largo, ya que puede que tarde un cierto tiempo.

```
5000 DIM k$(110,4): DIM
     k(110): DIM m(110): LET
     h$="0123456789ABCDEF":
     LET b$="": LET
     g$="0123456789abcdef"
5010 DIM t$(100,24): DIM
     r(100): DIM z$(100,6):
     DIM z(100)
5020 DIM b(9): LET b(1)=1:
     FOR i=2 TO 9: LET
     b(i)=b(i-1)+b(i-1): NEXT
     i
5030 DIM r$(8,4,4): FOR j=1
     TO 4: FOR i=1 TO 8:
     READ r$(i,j): NEXT i: NEXT
     j
5040 DATA "0","1","2","3",
     "4","5","6","7","nz","z",
     "nc","c","po","pe","p",
     "m","0","8","16","24",
     "32","40","48","56",
     "hl","ix","iy","bc","de",
     "hl","sp"," "
5050 DIM s$(8,2,4): DIM t(18):
     DIM u$(18,10): FOR j=1
     TO 2: FOR i=1 TO 8/j:
     READ s$(i,j): NEXT i: NEXT
     j: FOR j=1 TO 18: READ
     t(j),u$(j): NEXT j
5060 DATA "b","c","d","e",
     "h","l","(hl)","a","bc",
     "de","hl","sp",235,"de",
     8,"af",277,"(sp)",60742,
     "0",60758,"1",60766,
     "2",233,"(hl)",56809,
     "(ix)",65001,"(iy)",10,
     "(bc)",26,"(de)",60767,
     "r",2,"(bc)",18,"(de)",
     60751,"r",249,"sp",
     60743,"i",60759,"i"
5070 DEF FN b(x,i)=INT (x/
     b(i+1))-INT (x/b(i+2))*2
5080 DEF FN x(x,
     i)=x-b(i+2)*FN b(x,
     i)+b(i+1)
```




```
5090 DEF FN j(x,i)=INT
     x-b(i+1)*INT (x/b(i+1))
5100 DEF FN e(i$,j$)=(i$=j$(
     TO LEN (i$)))
5110 FOR i=1 TO 110: READ
     k$(i),k(i),m(i): IF NOT FN
     e("*",k$(i)) THEN NEXT i
5120 DATA "ld",10,10,"ld",26,
     10,"ld",60767,10,"ld",
     60759,10,"ld",2,138,
     "ld",18,138,"ld",60751,
     138,"ld",60743,138,"ld",
     64,22,"ld",50,202,"ld",
```

- TRADUCCION DEL LENGUAJE ENSAMBLADOR A CODIGO MAQUINA
- ETIQUETAS
- ESPACIO PARA VARIABLES Y DATOS
- FUNCIONAMIENTO DEL PROGRAMA
- PONIENDOLO A PRUEBA
- LOCALIZACION DE ERRORES
- PROGRAMA TRAZADOR

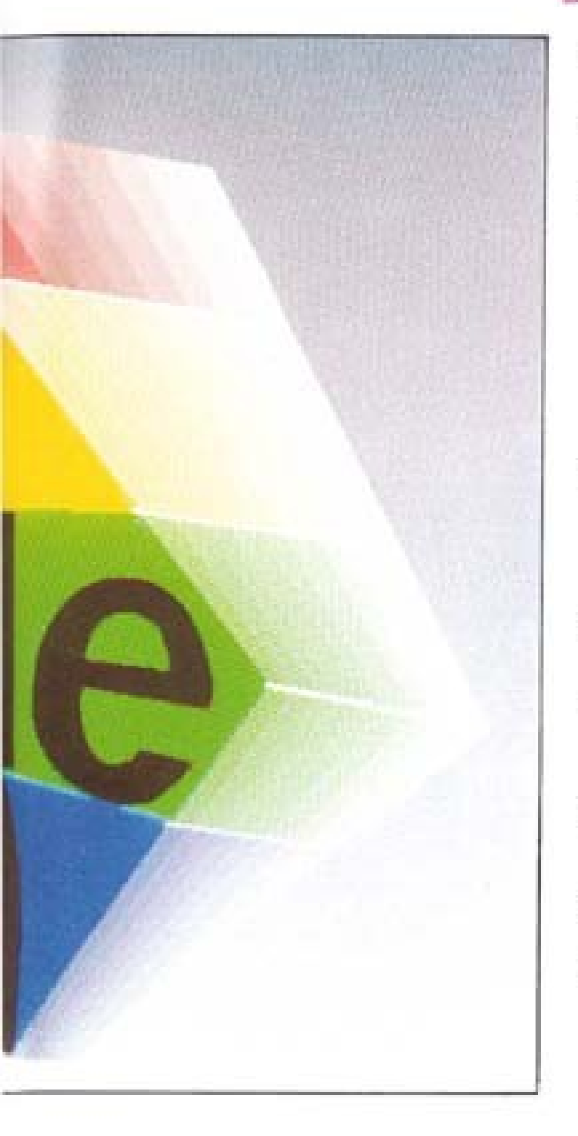


```
      58,74,"ld",249,139,"ld",
      34,195,"ld",42,67,"ld",
      6Ø779,197,"ld",6Ø771,
      199,"ld",97,165,"ld",64,
      54
513Ø DATA "adc",136,5Ø,
      "adc",6Ø746,3,"add",
      128,5Ø,"add",9,149,
      "and",16Ø,48,"or",176,
      48,"xor",168,48,"nop",Ø,
      Ø,"sub",144,48,"sbc",
      152,5Ø,"sbc",6Ø738,3,
      "cp",184,48,"jp",13Ø,45,
      "jp",233,9,"jp",568Ø9,9,
      "jp",65ØØ1,9,"jp",131,
      49,"jr",96,45,"jr",88,41
514Ø DATA "call",132,45,
      "call",141,41,"ret",2Ø1,
      Ø,"djnz",74,4Ø,"dec",11,
      17,"dec",5,16,"inc",3,17,
      "inc",4,16,"push",197,
      17,"pop",193,17,"di",
      243,Ø,"ei",251,Ø,"halt",
      118,Ø,"ex",235,139,"ex",
      8,15,"ex",227,143,"exx",
      217,Ø
515Ø DATA "rst",199,132,
      "rts",192,5,"bit",52Ø32,
      2Ø,"defb",-256,4Ø,"ccf",
      63,Ø,"scf",55,Ø,"cpl",47,
      Ø,"cpd",6Ø841,Ø,"cpdr",
      6Ø857,Ø,"cpi",6Ø833,Ø,
      "cpir",6Ø849,Ø,"daa",39,
      Ø,"im",6Ø742,8,"im",
      6Ø758,8,"im",6Ø766,8
516Ø DATA "in",6Ø736,13Ø,
      "in",149,42,"ind",6Ø848,
      Ø,"indr",6Ø81Ø,Ø,"ini",
      6Ø84Ø,Ø,"ldd",6Ø84Ø,Ø,
```

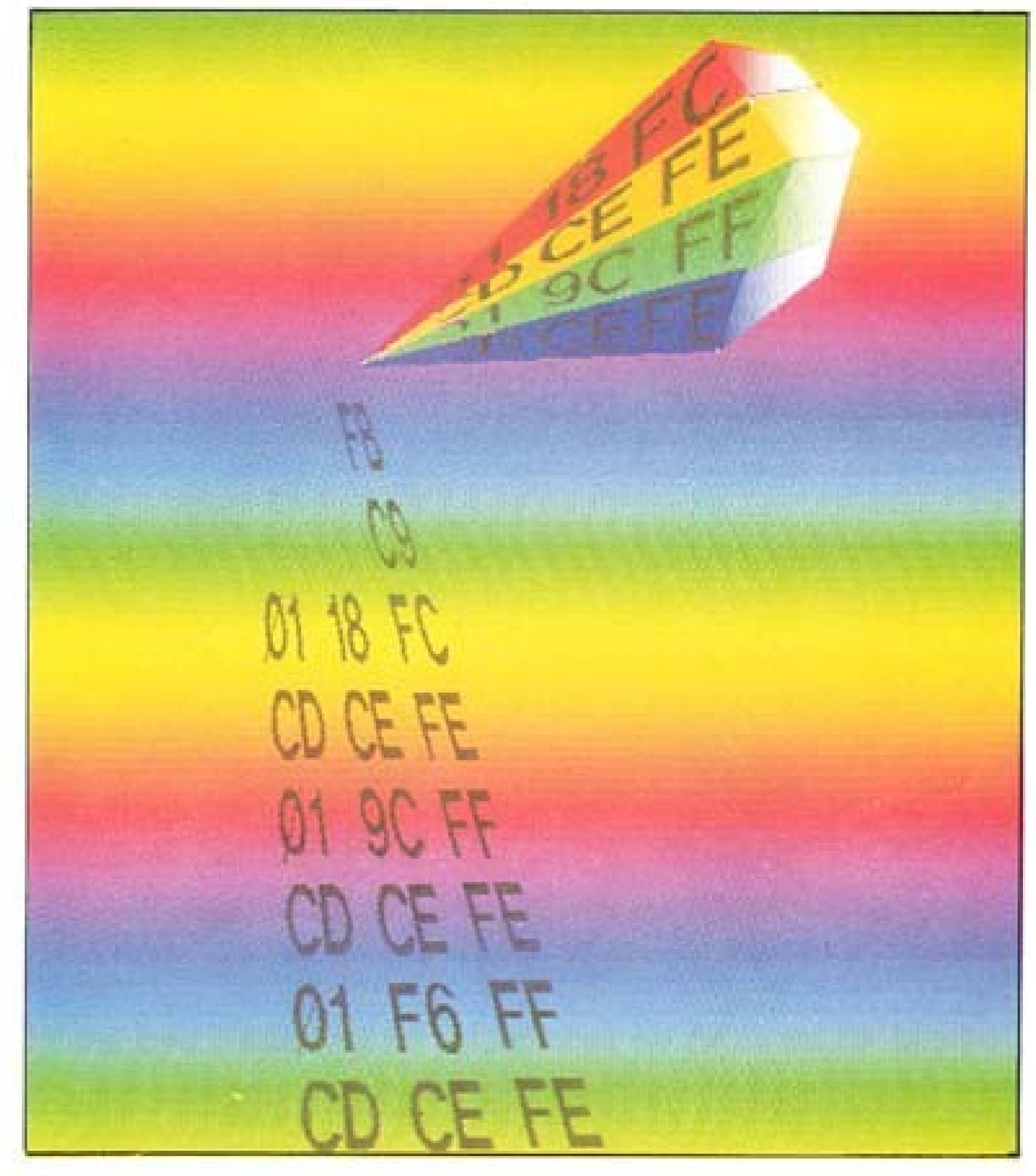
```
      "lddr",6Ø856,Ø,"ldi",
      6Ø832,Ø,"ldir",6Ø848,Ø,
      "neg",6Ø74Ø,Ø,"otdr",
      6Ø859,Ø,"otir",6Ø851,Ø,
      "out",6Ø737,2,"out",141,
      17Ø,"outd",6Ø843,Ø,
      "outi",6Ø835,Ø
517Ø DATA "res",52Ø96,2Ø,
      "reti",6Ø749,Ø,"retn",
      6Ø741,Ø,"rl",51984,64,
      "rla",23,Ø,"rlc",51968,
      16,"rlca",7,Ø,"rld",
      6Ø783,Ø,"rr",51992,64,
      "rra",31,Ø,"rrc",51976,
      16,"rrca",15,Ø,"rrd",
      6Ø775,Ø
518Ø DATA "set",5216Ø,2Ø,
      "sla",52ØØØ,16,"sra",
      52ØØ8,16,"srl",52Ø24,
      16,"defw",-256,41
519Ø DATA "*",Ø,Ø: LET ii=i:
      LET k(11Ø)=ii
52ØØ LET b=Ø: LET ba=PEEK
      23635+256*PEEK
      23636+4: LET n=1
521Ø LET cc=1: IF PEEK
      ba<>234 THEN LET
      n=n-1: GO TO 525Ø
522Ø LET cc=cc+1: LET
      ba=ba+1: IF PEEK ba=13
      THEN LET ba=ba+5: LET
      n=n+1: GO TO 521Ø
523Ø LET t$(n,cc)=CHR$ PEEK
      ba
524Ø GO TO 522Ø
525Ø FOR g=1 TO 1ØØ: LET
      r(g)=g-1: NEXT g: LET
      fh=1ØØ
53ØØ LET kØ=Ø: LET k9=99:
      LET pØ=Ø: LET vv=Ø
531Ø LET k=kØ: LET p=pØ
532Ø GO SUB 8ØØØ
533Ø GO SUB 7ØØØ: LET o$=i$:
      IF o$(1)="*" THEN PRINT
      o$;: GO TO 532Ø
534Ø IF o$="end" THEN PRINT
      "    end last addr ";p-1
535Ø IF o$="end" THEN LET
      pØ=Ø: GO TO 9999
537Ø IF o$<>"org" THEN GO
      TO 54ØØ
538Ø GO SUB 7ØØØ: LET s=Ø:
```




```
      IF i$(1)="*" THEN LET
      s=p: LET i$=i$(2 TO )
539Ø LET p=VAL i$+s: PRINT "
      org ";p;: GO TO 532Ø
54ØØ IF p=Ø THEN PRINT "(you
      forgot org)": LET p=5ØØØØ
541Ø LET p$=o$+"!": FOR
      i=1+18*(o$<>"ld") TO
      11Ø: IF o$<=k$(i) AND
      p$>k$(i) THEN GO TO
      55ØØ
542Ø NEXT i: PRINT o$
543Ø IF i$(1)="." THEN LET
      i$=i$(2 TO )
544Ø GO SUB 9ØØØ: LET
      gg=r(g)
545Ø IF gg<=1ØØ THEN LET
      s=SGN z(gg): LET b=INT
      (ABS z(gg)/65536): LET
      r=ABS z(gg)-b*65536:
      LET q=PEEK r+256*PEEK
      (r+1): PRINT " poking ";r;
      " with ";FN j(p*s+q,8): IF
      b THEN POKE (r+1),FN
      j((p*s+q)/256,8): PRINT "
      poking ";r+1;" with ";FN
      j((p*s+q)/256,8)
548Ø PRINT " (this line not
      recognised)"
549Ø GO TO 542Ø
55ØØ LET z=Ø: LET r=Ø: LET
      e=Ø: PRINT "      ";o$
551Ø LET op=k(i): IF m(i)=Ø
      THEN GO TO 6Ø9Ø
552Ø GO SUB 7ØØØ: LET a$=i$:
      PRINT " ";a$
553Ø LET m=m(i): LET op=k(i):
      LET b=FN b(m,Ø): LET
      b7=b+2*FN b(m,7)+1:
      LET z=Ø: IF FN j(m,3)<2
      THEN LET c$=a$: GO TO
      572Ø
554Ø FOR j=1 TO LEN a$: IF
      a$(j)="," THEN GO TO
      558Ø
555Ø NEXT j: IF o$="rst" OR
```




```
      o$="rts" THEN GO TO
      558Ø
556Ø IF FN e(o$,k$(i+1)) THEN
      LET i=i+1: GO TO 553Ø
557Ø PRINT " (two operands
      expected)": GO TO 532Ø
558Ø LET b$=a$( TO j-1): LET
      c$=a$(j+1 TO )
559Ø IF FN b(m,2) THEN GO TO
      565Ø
56ØØ IF FN b(m,7) THEN LET
      d$=c$: LET c$=b$: LET
      b$=d$
561Ø IF b$="ahl"(b+1 TO
      b+b+1) THEN GO TO
      572Ø
562Ø IF b$="(c)" AND (o$="in"
      OR o$="out") THEN GO
      TO 572Ø
563Ø IF (FN e(o$,k$(i+1))) AND
      (FN j(m(i+1),3)>=2)
      THEN LET i=i+1: GO TO
      553Ø
564Ø PRINT "(first operand a or
      hl expected)": GO TO 532Ø
565Ø IF FN b(m,1) THEN GO TO
      569Ø
566Ø LET e$=(b$+"□□□") ( TO
      4): FOR i=1 TO 8: IF
      e$=r$(j,b7) THEN LET
      op=op+8*(j-1)*(b7<4)
      +16*(j-6)*(b7=4)*(j>3):
      LET z=(j-1)*(b7=4)*
      (j<=3): GOTO 571Ø
567Ø NEXT j: IF p$>k$(i+1)
      AND (FN j(m(i+1),3)>=2)
      THEN LET i=i+1: GO TO
      553Ø
568Ø PRINT "(first operand bit or
      flag reqd)": GO TO 532Ø
569Ø IF FN b(m,7) THEN LET
      d$=c$: LET c$=b$: LET
      b$=d$: GO TO 566Ø
57ØØ LET x=8: GO SUB 575Ø:
      IF e THEN GO TO 573Ø
571Ø IF c$="" THEN GO TO
      6Ø9Ø
572Ø LET x=1+15*b+7*(op<=
      6 AND op>=4 OR
      b$="(c)"): LET b$=c$: GO
      SUB 575Ø: IF NOT e THEN
      GO TO 6Ø9Ø
573Ø IF e=2 OR p$>k$(i+1)
      AND FN j(m(i+1),3)=FN
      j(FN x(m,Ø),3) THEN LET
      e=Ø: LET i=i+1: GO TO
      553Ø
574Ø GO TO 532Ø
575Ø LET r=Ø: IF FN b(m,4)
      AND FN e("("( TO NOT b),
      b$) THEN LET z2=FN
      e("ix",b$(2-b TO )+"
      ")+2*FN e("iy",b$(2-b TO
      )+" "): IF z2 THEN LET
      z=z2: LET e$=b$( TO LEN
      b$-NOT b): LET
      b$="(hl)"(1+b TO 4-b):
      LET f$="Ø"+e$(4-b TO )
576Ø IF FN b(m,3) THEN GO TO
      579Ø
577Ø LET e$=(b$+"    ")( TO
      4): FOR j=1 TO 8/(b+1):
      IF e$=s$(j,b+1) THEN
      LET op=op+(j-1)*x:
      RETURN
578Ø GO TO 581Ø
579Ø LET j2=9+9*(o$="ld"):
      FOR j=j2-8 TO j2: IF
      k(i)<>t(j) THEN GO TO
      581Ø
58ØØ IF FN e(b$,u$(j)) THEN
      RETURN
581Ø NEXT j: IF b$="af" THEN
      IF FN e("p",o$) THEN LET
      op=op+48:
      RETURN
582Ø IF FN b(m,6) AND FN
      e("(",b$) THEN LET
      b$=b$(2 TO LEN b$-1):
      GO TO 586Ø
583Ø IF FN b(m,5) THEN LET
      op=FN x(op+6*NOT b,6):
      GO TO 586Ø
584Ø IF p$>k$(i+1) THEN LET
      e=2: RETURN
585Ø PRINT "(cannot match
      operand to op)": LET e=1:
      RETURN
586Ø LET r=65536
587Ø LET s=1
588Ø IF b$="" THEN GO TO
      6Ø8Ø
589Ø LET x$=b$(1): LET
      d$=b$(2 TO ): IF x$="*"
      THEN LET r=r+p*s: LET
      b$=d$: GO TO 587Ø
59ØØ IF x$="+" THEN LET
      b$=d$: GO TO 588Ø
591Ø IF x$="-" THEN LET
      b$=b$: LET s=-s: GO TO
      588Ø
592Ø IF x$="" THEN LET
      r=r+CODE d$*s: LET
      b$=d$(2 TO ): GO TO
      587Ø
593Ø LET q=Ø: IF x$<>"%" OR
      d$<"Ø" OR d$>="2"
      THEN GO TO 596Ø
594Ø IF d$>="Ø" AND d$<"2"
      THEN LET q=q*2+CODE
      d$-48: LET d$=d$(2 TO
      ): GO TO 594Ø
595Ø LET r=r+q*s: LET b$=d$:
      GO TO 587Ø
596Ø IF x$<>"$" OR d$<"Ø"
      OR d$>="g" THEN GO TO
      6ØØØ
```




```
5970 LET x$=CHR$ CODE d$:
     FOR g=Ø TO 15: IF
     x$<>h$(g+1) AND
     x$<>g$(g+1) THEN GO
     TO 599Ø
598Ø LET q=q*16+g: LET
     d$=d$(2 TO ): GO TO
     597Ø
599Ø NEXT g: LET r=r+q*s: LET
     b$=d$: GO TO 587Ø
6ØØØ IF x$<"a" OR x$>"z"
     THEN GO TO 6Ø4Ø
6Ø1Ø LET i$=b$: GO SUB 9ØØØ:
     IF i$<>"" THEN GO SUB
     94ØØ
6Ø2Ø IF r(g)<>23ØØØ AND
     r(g)>1ØØ THEN LET
     r=r+(r(g)-1ØØ)*s: LET
     b$=i$: GO TO 587Ø
6Ø3Ø IF r(g)=23ØØØ OR
     r(g)<=1ØØ THEN LET
     gh=r(fh): LET r(fh)=r(g):
     LET r(g)=fh: LET fh=gh:
     LET z(r(g))=(p+SGN
     op+(ABS op>255)
     +2*(z>Ø)+65536*((b OR
     FN b(m,6)) AND
     o$<>"jr"))*s: LET b$=i$:
     GO TO 587Ø
6Ø4Ø IF x$<"Ø" OR x$>"9"
     THEN LET r=Ø: GO TO
     6Ø7Ø
6Ø5Ø IF b$>="Ø" AND b$<":"
     THEN LET q=q*1Ø+CODE
     b$-48: LET b$=b$(2 TO
     ): GO TO 6Ø5Ø
6Ø6Ø LET r=r+s*q: GO TO 587Ø
6Ø7Ø PRINT "(address not
     understood)"
6Ø8Ø LET r=r-(p+2)*(o$=
     "djnz" OR o$="jr"):
     RETURN
6Ø9Ø PRINT TAB 16;: LET by=p/
     256: GO SUB 619Ø: LET
     by=p: GO SUB 619Ø: GO
     SUB 616Ø
61ØØ IF z THEN LET
     by=189+z*32: GO SUB
     618Ø: GO SUB 616Ø
611Ø IF op>=Ø THEN LET
     by=op/256: GO SUB
     617Ø: GO SUB 615Ø: LET
     by=op: GO SUB 618Ø: GO
     SUB 615Ø
612Ø IF r=Ø THEN GO TO 532Ø
613Ø GO SUB 616Ø: LET by=r:
     GO SUB 618Ø: IF (b OR
     FN b(m,6)) AND o$<>"jr"
     THEN LET by=r/256: GO
     SUB 618Ø
614Ø GO TO 532Ø
615Ø IF z AND INT by AND NOT
     b THEN GO SUB 616Ø:
     LET by=VAL f$: GO SUB
     618Ø: LET z=Ø
616Ø PRINT "  ";: RETURN
617Ø IF INT by<=Ø THEN
     RETURN
618Ø LET by=FN j(by,8): POKE
     p,by: LET p=p+1
619Ø LET by=FN j(by,8): PRINT
     h$(1+INT (by/16));h$(FN
     j(by,4)+1);
62ØØ RETURN
7ØØØ IF k>n THEN LET
     i$="end": RETURN
7Ø1Ø LET k1=k9+1: IF
     k9>=LEN t$(k) THEN LET
     i$="/missing/": RETURN
7Ø2Ø LET k9=k1: IF t$(k,k1)="
     " THEN GO TO 7Ø1Ø
7Ø3Ø IF k9>LEN t$(k) THEN
     LET i$=t$(k)(k1 TO ):
     RETURN
7Ø4Ø IF t$(k,k9)<>" " THEN
     LET k9=k9+1: GO TO
     7Ø3Ø
7Ø5Ø LET i$=t$(k)(k1 TO
     k9-1): RETURN
8ØØØ IF k>Ø THEN IF t$(k)(k9
     TO )>t$(99) THEN PRINT
     t$(k)(k9 TO );
```

```
8010 POKE 23692,0: LET
     k=k+1: LET k9=0
8020 PRINT : RETURN
9000 LET x$=""
9010 IF i$<"a" OR i$>"z"
     THEN GO TO 9030
9020 LET x$=x$+i$(1): LET
     i$=i$(2 TO ): GO TO 9010
9030 IF i$<>"" THEN RETURN
9400 FOR g=1 TO vv: IF FN
     e(x$,z$(g)) THEN RETURN
9410 NEXT g: LET vv=vv+1:
     LET z$(vv)=x$: LET g=vv:
     LET r(g)=23000
9420 RETURN
```

FUNCIONAMIENTO

Habrás observado que el programa ensamblador empieza en la línea 5000. Se hace así para dejar sitio a tus programas en lenguaje ensamblador, que han de introducirse como sentencias REM en las líneas del BASIC.

Tienes que introducir cada instrucción en lenguaje ensamblador como una línea separada, en una sentencia REM separada y con letras minúsculas; no uses mayúsculas.

Antes de ensamblar el programa, tienes que borrar una zona mediante CLEAR para que pueda ir allí tu código máquina. La primera línea de tu programa en lenguaje ensamblador debe ser algo parecido a:

```
10 REM org 32000
```

32000 es la dirección de memoria donde empieza el código máquina y naturalmente estará sobre la zona abarcada por tu borrado previo.

Si se te olvida especificar org, que es el origen, el ensamblador se irá por defecto a la posición 50000 y allí se quedará tu programa.

Se utilizan los mnemónicos corrientes del *Z80* con una excepción: los retornos condicionales. Al final de las rutinas de lenguaje ensamblador del *Z80* el mnemónico *ret* sirve para hacerte volver al cuerpo principal del programa en código máquina. En este ensamblador funcionará el mnemónico *ret*.

Sin embargo, algunas veces puede ser que quieras hacer un retorno condicional, por ejemplo *ret nz*. Esto significa: volver si el resultado no es cero. Con este ensamblador deberás usar la sintaxis *rts nz*. Tienes que usar *rts* en vez de *ret* en todos los retornos condicionales, es decir, en cualquier retorno que contenga letras a continuación de *ret*. Para los retornos incondicionales, en los que no figura nada después de *ret*, se mantiene la forma *ret*.

Si utilizas números hexadecimales, deben ir precedidos del signo $. Los números en binario deberán llevar delante un signo %. Si delante de un número no hay signo alguno, el ensamblador lo considerará como decimal. Cualquier palabra que no sea un comando se considera como una etiqueta. Evita el uso de cosas muy parecidas y no utilices números.

Los programas escritos en lenguaje ensamblador deben finalizar con *REM end*, para que el ensamblador sepa cuándo tiene que pararse.

Una vez que hayas tecleado tu programa en lenguaje ensamblador, no tienes más que ejecutarlo tecleando RUN, con lo que te aparecerá en la pantalla un listado del programa —el código fuente— y su equivalente en código máquina —el código objeto—. Al mismo tiempo, dicho código objeto es POKEado en la memoria.

Si ahora te das cuenta de que has cometido un error en alguna de las líneas, puedes listar el programa y editar el lenguaje ensamblador de la forma habitual.

Una vez que esté ensamblado todo el código máquina, aparecerá en la pantalla la dirección final de la rutina en código máquina.

Para ejecutar el programa tienes que utilizar una de las instrucciones que se usan para hacer correr los programas en código máquina como RANDOMIZE USR.

Para almacenar el código objeto, teclea: SAVE «nombre» CODE dirección de comienzo, número de bytes.

Nombre, es el nombre que quieras darle tú a las rutinas. Debe ir entre comillas. La *palabra reservada* CODE informa al ordenador de que tiene que almacenar el programa byte a byte, a diferencia de un programa en BASIC, que puede ser reubicado en cualquier parte de la memoria. La dirección de comienzo es el origen del programa en código máquina. Puedes calcular el número de bytes restando la dirección de origen de la dirección final y sumando 1.

Tanto el ensamblador como el código fuente se pueden almacenar utilizando la rutina normal de SAVE.

Para comprobar tu ensamblador, prueba a teclear el programa en lenguaje ensamblador para hacer scrolling hacia la derecha, descrito anteriormente en INPUT. Bien sea que lo ensambles a mano o con ayuda de tu ensamblador, el código máquina resultante será:

```
11 FF 57 21 FE 57 06 C0 C5 1A
   01 1F 00 ED B8 12 2B 1B
   C1 10 F3 C9
```

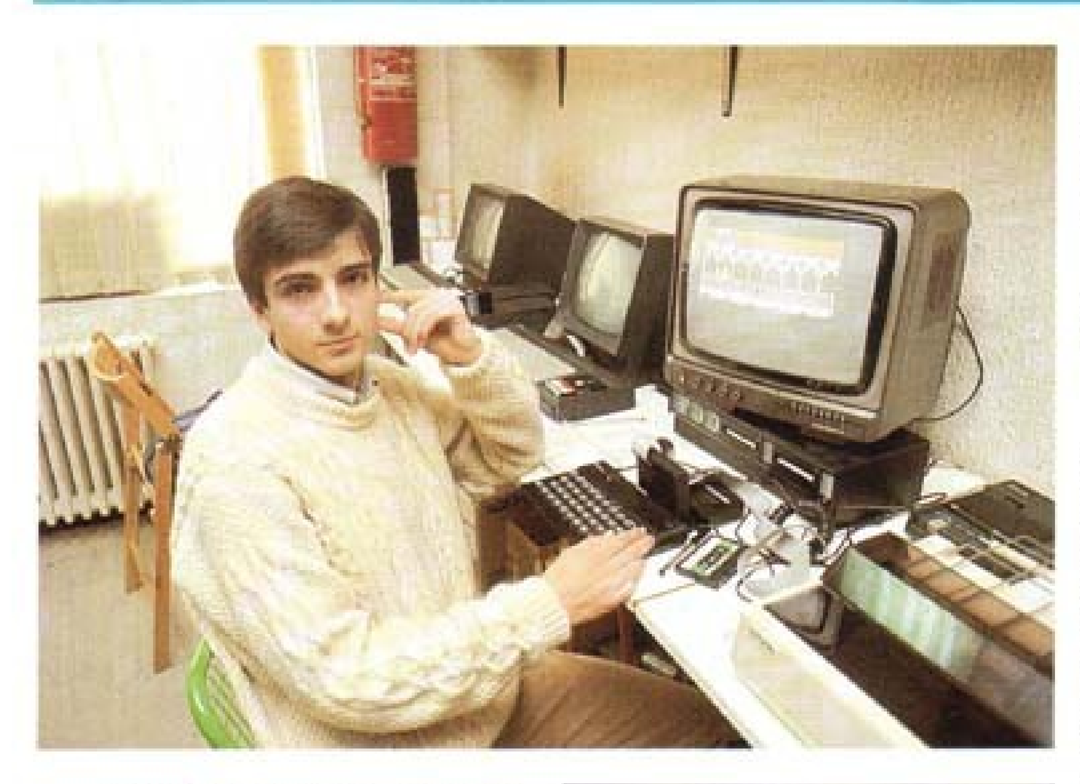
Observa que el lenguaje ensamblador ha de introducirse con letras minúsculas y no mayúsculas, ya que en caso contrario el ensamblador no los reconocería. Ahora ya puedes probar la rutina y ver cómo funciona.

LOCALIZACIÓN DE ERRORES

Cuando se teclea un programa largo, como el que acabamos de ver, resulta muy difícil no introducir errores. El error más corriente es olvidarse de introducir algún valor de DATA. Si recibes un mensaje de error que te diga «OUT OF DATA», comprueba tus sentencias DATA. Con una sola coma que te falte, ya puedes tener problemas. Si tu ensamblador no funciona a la primera no te desesperes. Si no consigues localizar un error, INPUT publicará para ti dentro de poco un programa trazador para el *Spectrum*, que te ayudará a encontrar los errores.

Un programa trazador presenta en la pantalla el número de la línea de BASIC que está siendo ejecutada en cada momento. El trazador que veremos presenta también el número de la sentencia que está siendo ejecutada en dicha línea. El uso de estas utilidades en unión del programa publicado hace que la detección de errores sea fácil.

HABLAMOS CON EL AUTOR DE «EL PUNKI DE AKI»



— ¿Cómo? ¿**José Carlos Arboiro**? Pues no, no le conozco. ¿Quién es?

— El autor de **Ramón Rodríguez** nada menos.

— ¿Quién? ¿El de *las desventuras de un punki de akí*?

— Sí hombre, sí. Ese mismo.

— ¿Pero cómo? ¿Es de *aquí* de verdad?

— Claro que sí, espabilao. Que no te enteras.

Cuando se habla del soft español, casi irremediablemente se acaba como los protagonistas de esta hipotética conversación. Pero, seamos sinceros, ¿A que te hemos pillado? ¿A que sabías perfectamente que **Ramón Rodríguez** es uno de los últimos grandes éxitos de ventas, y sin embargo no tenías ni la menor idea de quién era su autor? ¿Eh?

Pues bien, si no quieres que esto vuelva a ocurrirte, entonces quizás te convenga saber sobre **José Carlos Arboiro** algo más que su nombre. Para satisfacer tu curiosidad, nos fuimos a hacerle una visita con una larga lista de preguntas debajo del brazo.

LA ENTREVISTA

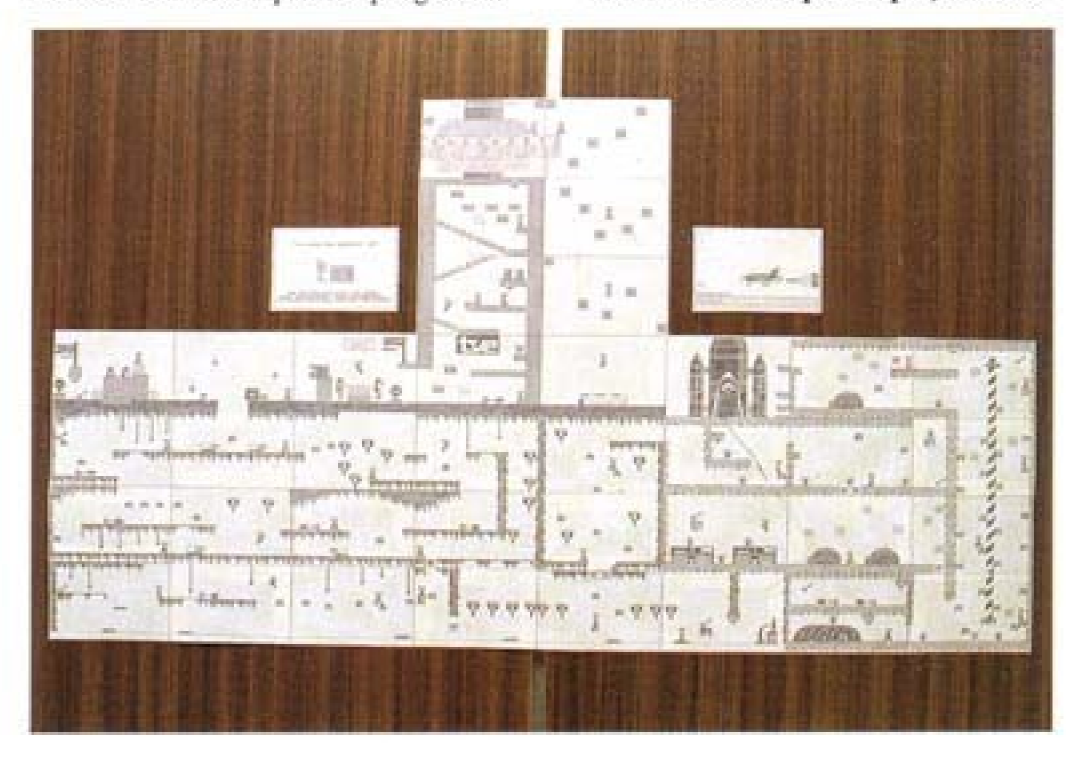
Nos recibió en las oficinas de **ERBE**, firma para la que actualmente trabaja, rodeado por un sofisticado equipo con el que no podía ni soñar cuando realizó su primer programa.

Después de las presentaciones de rigor, instalamos nuestros artilugios de tortura y comenzamos un largo interrogatorio, que se prolongaría durante más de una hora.

INPUT: José Carlos, ¿cuándo empezaste a dedicarte en serio a la programación?

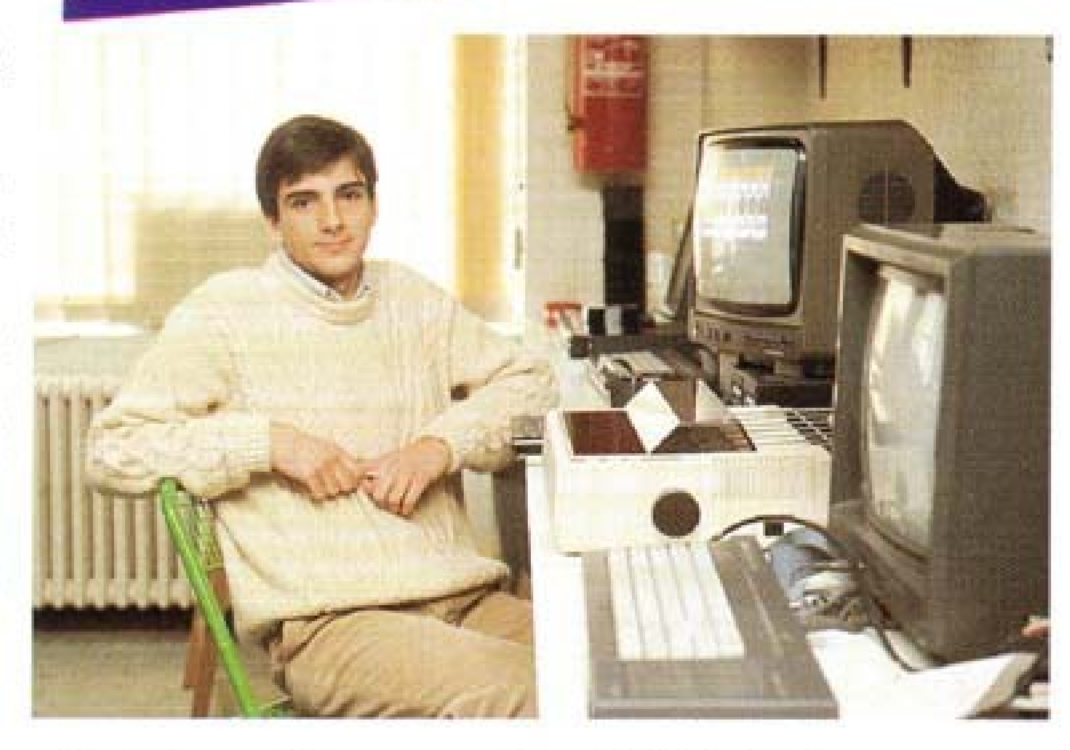
JCA: Empecé en el colegio, hace cuatro o cinco años, programando juegos educativos. Al cabo de un tiempo, robaron todo el equipo que allí teníamos y no tuve más remedio que olvidarme del tema por una temporada, hasta que me compré un Spectrum y empecé a programar por mi cuenta. El año pasado, después de un intenso trabajo durante todo el verano, vio la luz **Ramón Rodríguez**. Puede decirse que éste fue mi primer trabajo *serio*. En cuanto lo terminé, me puse en contacto con **ERBE**, les gustó la idea, y antes de que pudiera darme cuenta el programa ya se estaba vendiendo.

INPUT: Es de suponer que, como to-



dos los buenos programadores que tenemos en España, que no son pocos, eres autodidacta.

JCA: Pues sí. Aprendí BASIC con un ZX-81 y algunas revistas, de las pocas que había entonces, pero este lenguaje pronto se me quedó pequeño. Fue entonces cuando empecé a interesarme por algo misterioso que se llamaba *Código Máquina*. Me compré un libro sobre el tema, y poco a poco fui aprendiendo. Como entonces no había ensambladores, tenía que introducir las rutinas pokeando pacientemente, o usando listados REM, pero a pesar de las dificultades pude seguir adelante.



INPUT: Sé que no te gusta hablar del tema, pero la pregunta es obligada: ¿Qué opinas de la piratería?

JCA: Supongo que todos los programadores tenemos perfectamente asumido el problema. Sabemos que los piratas son personas que se aprovechan impunemente de nuestro trabajo, y que en la mayoría de los casos obtienen de sus actividades ilegales mucho más dinero del que nosotros ganamos con nuestro esfuerzo, pero no hay más remedio que aceptarlo. Espero que con la nueva legislación se dé un importante paso adelante en la lucha contra la piratería, aunque está muy claro que con eso no basta.

INPUT: ¿Qué solución propones tú para combatir la piratería?

JCA: Yo creo que si las distribuidoras bajaran los precios, los usuarios que ahora se ven obligados a comprar copias piratas por no poder pagar un original, cambiarían de actitud. A nadie le importaría pagar un poco más para obtener un original, en vez de una copia de mala calidad sin ninguna garantía. El problema es que muchos usuarios piensan que los programas baratos son peores, y que su calidad depende del dinero que cuestan. Por eso, esta solución sólo sería efectiva si todas las distribuidoras se pusieran de acuerdo, y los usuarios supieran corresponder al esfuerzo.

(Sin duda, uno de los rasgos más destacados de la personalidad de José Carlos es la modestia. Llegados a este punto en la conversación, nos comentó, sin darle la menor importancia a sus palabras, que compaginaba la programación con sus estudios de C.O.U. y con el atletismo.

Antes de pasar a hablar de sus proyectos, le hicimos algunas preguntas sobre **Ramón Rodríguez.**)

INPUT: ¿Cómo nació la idea de tu programa?

JCA: La verdad es que nació mucho tiempo atrás, cuando programaba en el colegio. El personaje entonces era más alto, y aún no tenía mucho que ver con el actual. Más tarde lo convertí en un punki *playero*, pero finalmente deseché la idea. Mis programas sufren muchos cambios durante su elaboración porque no sigo un plan previo, sino que voy incorporando ideas a medida que se me van ocurriendo. De esta forma el producto final es mucho más original e imaginativo.

INPUT: Nos han llegado voces que aseguran que preparas un sensacional programa ambientado en la ciudad de Venecia...

JCA: Sí, pero por el momento no puedo deciros nada.

INPUT: Anda, sé bueno...

Después de unos minutos de súplicas y promesas, conseguimos arrancar de su generosidad algunos datos.

El programa reproducirá fielmente en sus escenarios los más pintorescos rincones de la ciudad de **Venecia** («hasta en sus más mínimos detalles»). El argumento se plantea como un arcade de persecución, aunque también habrá algo de estrategia. El protago-







HARD-SOFT-ACTUALIDAD

nista de la aventura es un apuesto galán veneciano, cuya misión todavía no está muy clara. Sobre el resto de los personajes tampoco hay nada seguro, pero José Carlos nos comentó que probablemente habrá un «cornudo con muletas» incordiando en algunas pantallas. Por otra parte, el comportamiento de los personajes secundarios, «los malos», será inteligente.

¿Que cuándo estará listo para su publicación? Probablemente no antes de la primavera o principios del verano. Esperamos con impaciencia.

Afortunadamente, durante la entrevista, nadie descubrió la mini-cámara fotográfica que uno de nosotros ocultaba en la solapa. Cuando llegamos a la redacción, corrimos atropelladamente al laboratorio y nos encerramos en él durante media hora de duro trabajo con nuestra destartalada ampliadora. Las lágrimas empañaron nuestros ojos: ¡¡teníamos las fotografías!!

¡Ah! No se lo digas a nadie.

PANTALLA GRANDE

Acaban de publicarse en el Reino Unido cinco programas basados en las películas más taquilleras de los últimos meses, todas ellas recientemente estrenadas en España. Se trata de LABERINTO, ALIENS, GOLPE EN LA PEQUEÑA CHINA, HOWARD y SHORT CIRCUIT, que vienen a unirse a LOS GOONIES, TOP GUN, COBRA y varios títulos más aparecidos durante las Navidades. Parece que las compañías productoras de soft no quieren renunciar al filón de los éxitos de la pantalla grande, a pesar de que muchos usuarios que no se dejan engañar por un título conocido empiezan a cansarse. Esperamos que pronto acabe la racha, y podamos volver a ver originalidad en nuestras pequeñas pantallas, con tramas y héroes inéditos.

****

SOFT-HARD EN LA RADIO

Desde hace algún tiempo, la cadena COPE viene emitiendo todos los sábados de cinco a siete un programa dedicado al mundo de la microinformática, con entrevistas, concursos, descripción de novedades en soft y hard, y una participación muy directa de los oyentes. Como siempre, en todo lo que a actualidad se refiere, la radio está por delante. Ojalá cunda el ejemplo.

****

DOS CONTRA DOS

Posiblemente recuerdes aquel programa de baloncesto que revolucionó los juegos de tema deportivo, hace ya más de un año, y que se llamaba ONE ON ONE. Pues bien, está a punto de aparecer en

GUIA DEL COMPRADOR DE SOFT

Es posible que este mes eches en falta alguno de estos títulos que, invariablemente, presiden todas las listas allá donde las haya. La verdad es que esos que faltan aún siguen cosechando éxitos, pero después de un largo año de gloria no queda más remedio que dejar paso a las novedades. Renovarse o morir.

TITULO	CALIFICACIÓN (1 a 10)
ANTIRIAD	9
FIRELORD	9
HEARTLAND	9
GREAT ESCAPE	9
FROST BYTE	8
PYRACUSE	8
STRIKE FORCE	
HARRIER	8
ACE	8
PRODIGY	8
PAPER BOY	8
BACK TO SKOOL	7
STAINLESS STEEL	7
JACK THE NIPPER	7
URIDIUM	7
BREAK THRU	7
CRYSTAL CASTLES	7
ASTERIX	7
FAIRLIGHT II	7
WAY OF THE TIGER	7
AVENGER	7
ARMY MOVES	7
DRAGON'S LAIR	7

TITULO	CALIFICACIÓN (1 a 10)
NUCLEAR BOWLS	7
WINTER GAMES	7
GHOST'N GOBLINS	7
FUTURE KNIGHT	7
RAMON RODRIGUEZ	6
GOONIES	6
MIAMI VICE	6
MERMAID MADNESS	6
LA JOYA DEL NILO	6
ROCKMAN	6
BOBBY BEARING	6
NIGHTMARE RALLY	6
STREET HAWK	6
EQUINOX	6
PING-PONG	6
BOUNDER	6
CAULDRON II	6
COBRA	6
DYNAMITE DAN II	6
THANATOS	6
TENNIS	6
INFILTRATOR	6
MAILSTROM	5

NOVEDADES EN CANDIDATURA

Después de la gran avalancha de Navidad, pocas son las novedades aparecidas. También son pocos los títulos que han conseguido pasar a la lista de éxitos, y por ello programas que a continuación listamos repiten candidatura.

ALIENS
TOP-GUN
THEY SOLD A MILLION
SUPER SOCCER
COSA NOSTRA
LIVINGSTONE SUPONGO
DEACTIVATORS
HARD BALL
TRAILBLAZER
EXPLODING FIST II
REVOLUTION
SILENT SERVICE
BAZOOKA BILL
HIGHLANDER
SHORT-CIRCUIT
TRIVIAL PURSUIT
FAT WORM
HACKER II
DAN DARE
LABERINTO
GOLPE EN LA PEQUEÑA CHINA
HOWARD
DEEP STRIKE

España TWO ON TWO, un simulador con cuatro jugadores en pantalla, donde lo que prima no es la acción personal, sino el juego de equipo. Hemos visto algunas imágenes en su versión para otro modelo de ordenador, y nos ha parecido un poco deficiente en cuanto a los gráficos, pero por el momento no podemos deciros más.

****

CON LA BOCA ABIERTA

Con la boca abierta se han quedado los usuarios de Commodore al probar un programa que hará historia: WORLD GAMES (Juegos del mundo). Todas las revistas británicas han coincidido en afirmar que la última producción de EPYX, continuación de la serie SUMMER GAMES-WINTER GAMES (Juegos de verano-Juegos de invierno), es lo más cercano a la perfección absoluta que se ha visto hasta ahora en un microordenador.

Esperamos anhelantes que este magnífico juego esté disponible para Spectrum pronto, o la envidia nos corroerá las entrañas.

****

AVENTURAS Y DESVENTURAS DE UN PUNKI DE AKI

DATOS GENERALES	
TITULO	Ramón Rodríguez
FABRICANTE	Erbe
TEMA	Un punki en apuros

CALIFICACION (Sobre 10 ptos.)	
ORIGINALIDAD	9
INTERES	7
GRAFICOS	7
COLOR	7
SONIDO	8
TOTAL	38





Como hacía todas las mañanas antes de salir a la calle, metió los dedos en el enchufe y aguantó estoicamente la descarga. Con aire preocupado, volvió su agujereado rostro hacia el espejo y se colocó un imperdible en el párpado, mientras hacía esfuerzos sobrehumanos por no mirar hacia arriba. Desde hacía ya mucho tiempo, los doscientos veinte voltios de la instalación de su casa le dejaban los pelos un poco fláccidos para su gusto, y por no echarse a llorar prefería no verlos. Un buen amigo le había recomendado insistentemente la alta tensión como solución a su problema, pero Ramón era un chico prudente y tampoco quería pasarse.

Consternado por la escasa efectividad de su matutino tratamiento capilar, salió a la calle y se dirigió hacia la tertulia neo-punk de la cervecería BASURA'S, donde le estaría esperando la peña. Pero de pronto, tuvo la certeza de que algo marchaba mal: ¡se había perdido!

En ese preciso instante, dos monstruosos gusanos gigantes de las profundidades le cerraron el paso. Sin pensárselo dos veces, nuestro aguerrido punki saltó ágilmente sobre ellos, con tan poca fortuna que fue a parar a un profundo abismo. Durante la caída, tuvo la absurda idea de agitar los brazos como un pajarito «por si acaso funciona», y para su sorpresa, funcionó. «Cuando cuente esto en el bar, me llevan a *Alcohólicos Anónimos*.»

Buscando el modo de salir de aquel infierno, pasó por varios sitios que le resultaban familiares: el muro de Berlín, TVE, el Polo Norte... Por más que se esforzaba, no lograba recordar dónde los había visto antes. Cuando ya comenzaba a perder las esperanzas, encontró varias llaves, y algo le hizo sospechar que tenían alguna relación con la solución del enigma. ¿Cuál? ¡Quién sabe!



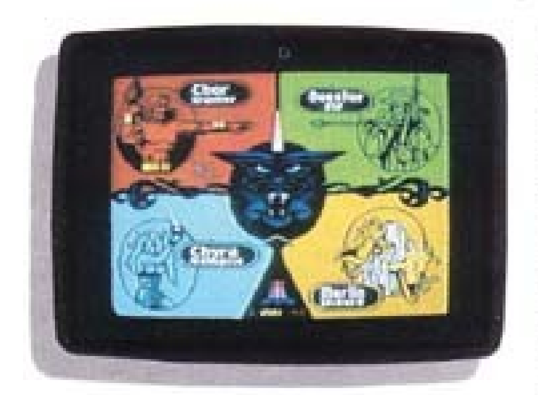


****

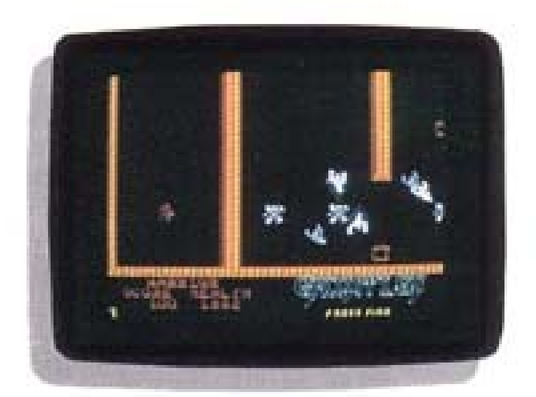
GAUNTLET

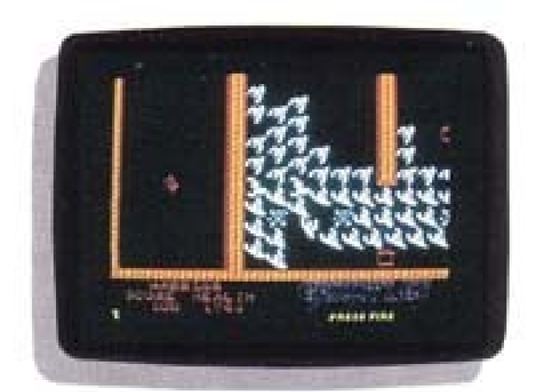
Recientemente, **U.S. GOLD** ha publicado en nuestro país la versión para Spectrum de **GAUNTLET**, una de las videoaventuras de mayor éxito en las máquinas tragaperras de todo el mundo.

El objeto del juego consiste simplemente en sobrevivir durante el mayor tiempo posible en el interior de un laberinto, enfrentándose a los monstruos que lo pueblan. Existen tres tipos distintos de armas, con diferente grado de efectividad: las armas arrojadizas, los conjuros, y las armas de combate cuerpo a cuerpo.



Cada una de ellas será especialmente poderosa contra determinados obstáculos o enemigos, y poco efectiva si se emplea sobre objetivos inadecuados.
Una de las características más innovadoras que presenta el programa, es la posibilidad de que participen simultáneamente en la aventura **dos jugadores**, bien sea con el joystick o con el teclado, guiando a personajes diferentes. Antes de comenzar el juego, tiene lugar la elección del héroe. Se puede escoger entre cuatro personajes, cada uno con sus propias características: Thor, el guerrero, posee unas dotes extraordinarias para el combate, pero sus poderes mágicos son escasos. El Mago Merlín, en cambio, está indefenso en la lucha cuerpo a cuerpo, pero tiene grandes poderes sobrenaturales. El elfo Questor, y la amazona Thyra, poseen aptitudes tanto para el combate como para la magia, pero ninguno de sus poderes destaca especialmente sobre los demás.
Para terminar, os diremos que GAUNTLET es un arcade en su estado más puro, una especie de sobredosis de acción, recomendable a todos aquellos que gusten de poner a prueba sus reflejos con un programa trepidante.





DATOS GENERALES

TITULO Gauntlet

FABRICANTE U.S. Gold

TEMA Arcade

CALIFICACION (Sobre 10 ptos.)

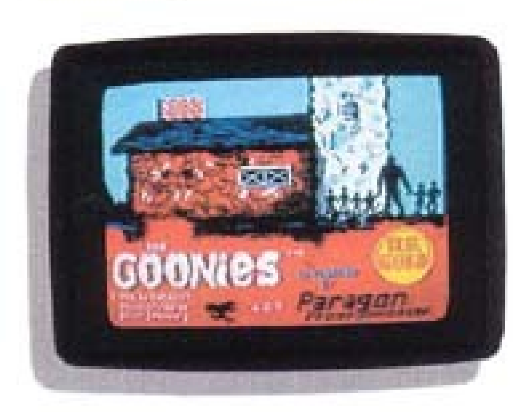
ORIGINALIDAD	7
INTERES	7
GRAFICOS	6
COLOR	6
SONIDO	7
TOTAL	33



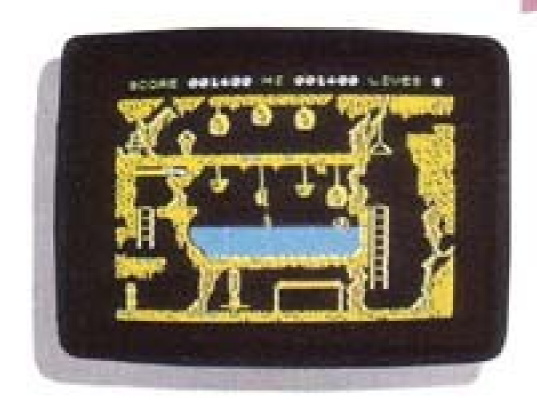
****

LOS GOONIES

Como ya habrás adivinado, se trata de la versión para *micro* de la película del mismo nombre, producida por *Warner Bros*, en 1986, y dirigida por uno de los grandes maestros del cine de fantasía y aventuras, **Steven Spielberg**.
En líneas generales, el argumento coincide fielmente con el tema original de la película:
La pandilla de los **Goonies** se encuentra en una situación difícil. Sus familias van a ser desahuciadas si no reúnen pronto una gran suma de dinero para pagar las hipotecas, y la única forma de conseguirlo es encontrando el tesoro perdido del Pirata Tuerto. Desgraciadamente, los temibles hermanos Fratelli tratarán de impedir por todos los medios que los Goonies alcancen su objetivo. Además, el camino que tendrán que recorrer estará sembrado de trampas y obstáculos, y poblado por seres de las profundidades, como calaveras volantes, murciélagos asesinos, y pulpos gigantes.









En todas las pantallas habrán de colaborar **dos** Goonies juntos para resolver el enigma y continuar la aventura. Cada vez que pierdas uno, otro ocupará su lugar, hasta que tu reserva de Goonies se agote y tengas que empezar de nuevo.
Pero debemos mencionar dos aspectos que no nos han gustado. El número de pantallas es muy reducido (ocho en total), y el grado de dificultad es tan bajo, que sería un crimen sugerir alguna pista.
No obstante, desde otro punto de vista se podrían valorar positivamente ambos aspectos, sobre todo teniendo en cuenta que a muchos usuarios no les gusta tener que romperse la cabeza pensando para resolver un juego, o que los más pequeños también tienen derecho a poder terminar por sí mismos la aventura.

DATOS GENERALES

TITULO The Goonies

FABRICANTE U.S. Gold

TEMA Aventuras

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	6
INTERES	7
GRAFICOS	7
COLOR	7
SONIDO	6
TOTAL	33

****

INFILTRATOR

Telegrama urgente para el capitán Johnny «Jimbo-Baby» McGibbits, más conocido como **Infiltrator**:
Asunto.— El líder loco del país vecino vuelve a hacer de las suyas.
«Necesitamos tu ayuda, Jimbo.»
«Sólo un piloto de helicópteros, experto en balística, ingeniero, neurocirujano, político, actor de cine, estrella del rock, motociclista, cinturón negro de kárate, y tío guaperas como tú, puede salvar al mundo.»
«Hay que detener al loco a cualquier precio.»
«Whizbang Enterprises nos ha cedido amablemente el último prototipo del super-helicóptero-turbopropulsado Gizmo-DHX-1. Con él deberás infiltrarte en las bases enemigas para cumplir las misiones que se te asignen.»
«Para facilitarte las cosas, te enviamos una edición de bolsillo de la "Guía McGibbits de la infiltración en instalaciones terrestres", y un cuaderno con todos los detalles referentes a la operación.»

DATOS GENERALES

TITULO Infiltrator

FABRICANTE U.S. Gold

TEMA Simulador/Arcade

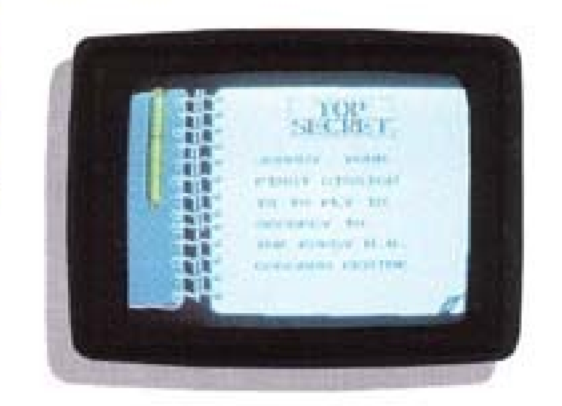


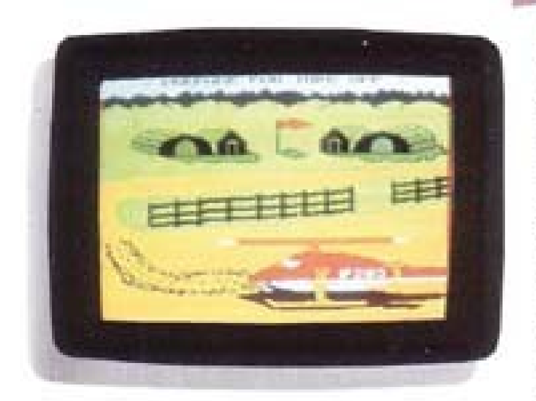
CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	8
INTERES	7
GRAFICOS	7
COLOR	7
SONIDO	6
TOTAL	35

«¡Suerte, Jimbo!»
En líneas generales, podemos afirmar que el programa cumple todos los requisitos esenciales para merecer una opinión muy favorable por nuestra parte. No obstante, debemos señalar que la versión para Spectrum presenta muchos aspectos negativos desde el punto de vista gráfico, especialmente si la comparamos con la que en su día se







hizo para Commodore. Todos esperábamos que se sacara mucho más partido a la versión original de **Infiltrator**, pero como suele ocurrir en estos casos, los resultados han sido un poco decepcionantes. No obstante, quedamos a la espera de que en un futuro cercano Aackosoft nos sorprenda gratamente con otro personaje, aunque no sea tan polifacético como este «infiltrado».



****

LA GRAN EVASION

Poco antes de que la gran avalancha navideña inundara el mercado, **OCEAN** presentó en España un programa que, desde entonces, no ha bajado de los primeros puestos en las listas de ventas, confirmando así el rotundo éxito obtenido en Inglaterra. Se trata de **THE GREAT ESCAPE**.

EXPERTO EN FUGAS

Pocos son los que pueden presumir de haber pasado unas agradables *vacaciones* en un campo de concentración alemán durante la Segunda Guerra Mundial. Esta aventura trata la sorprendente historia de un anónimo capitán de intendencia, cuyo nombre poco importa, que no sólo ha sido cliente asiduo de los peores campos de prisioneros de toda Europa, sino que además cuenta con el más brillante historia como **experto en fugas**.

...Después de numerosos intentos de fuga y varios traslados, acabó dando con sus molidos huesos en un viejo castillo austriaco, habilitado como prisión de máxima seguridad durante la guerra. Corría el año 1942.

El jefe al mando en aquel nuevo campo de internamiento era un remilgado oficial prusiano, que tenía por costumbre irrumpir por sorpresa en los lugares y momentos más inoportunos, creando no pocos problemas a quienes preparaban alguna fuga. En su desmedida afición por la disciplina y el orden, había impuesto desde su llegada un horario de actividades muy estricto, que apenas dejaba ratos libres. Por si esto fuera poco, reforzó la plantilla hasta conseguir que hubiera un guardián por cada prisionero, y dio órdenes de que se castigara severamente a todo aquel que se saliera de la rutina del campo, o abandonara las actividades obligatorias después del toque de sirena.

El recinto estaba rodeado por una alambrada doble, con torretas de vigilancia instaladas en todos los ángulos, y numerosos guardias patrullando en las inmediaciones. En la parte exterior del perímetro, mortíferos perros adiestrados para descuartizar a todo prisionero que se les pusiera por delante, daban buena cuenta de los pocos que se atrevían a intentar la fuga.

Sólo un plan audaz e imaginativo podría tener una remota posibilidad de éxito en un infierno como aquél. La única ventaja con que se podía contar, era la existencia de una importante organización interna dedicada a la preparación de fugas. Gracias a ella, no era muy difícil obtener herramientas, disfraces, documentos falsificados, y todo el material necesario para la huida. Además, la **Cruz Roja** enviaba periódicamente paquetes que ocultaban gratas sorpresas en sus dobles fondos.

...Cuando nuestro sufrido «experto

DATOS GENERALES

TITULO The great escape

FABRICANTE Ocean

TEMA Fuga del campo de concentración

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	10
INTERES	10
GRAFICOS	9
COLOR	7
SONIDO	7
TOTAL	43







en fugas» tuvo noticias de los túneles que sus compañeros habían construido, supo que el gran momento había llegado.

EL PLAN

Primer día:

Antes de acudir a la formación para pasar lista, recogió su paquete de la **Cruz Roja** en el botiquín. Con satisfacción comprobó que contenía una **bolsa** con casi todo lo necesario para la fuga. Después del desayuno, abrió con la **llave roja** la puerta de acceso a las habitaciones del ala Este, donde encontró **la linterna**. Allí dejó la llave, pues sabía que no podía llevar simultáneamente más de dos objetos, y se marchó apresuradamente con la linterna y la bolsa en su poder. Cuando volvió a su barracón después de los ejercicios, lo escondió todo en la entrada del **túnel uno**, ya que por allí realizaría la fuga.

Segundo día:

Salió al patio y buscó la **llave verde** junto a las patas de una de las torretas. Con ella logró entrar en el almacén, para coger las **herramientas** que le permitirían abrir la puerta contigua, donde se encontraba **la pala**. Cogió ésta, y volvió al túnel



para esconderla. Más tarde, casi olvida recoger de nuevo el paquete de la Cruz Roja, que en esta ocasión contenía una herramienta imprescindible: **las tenazas.**

Tercer día:

Se aburrió mucho esperando sin nada que hacer a que llegara el paquete del cuarto día. En toda la jornada, sólo tuvo que recoger la **pastilla de chocolate** que le envió la Cruz Roja y esconderla, para evitar que la volvieran a enviar.

Cuarto día:

Al abrir su paquete, encontró el instrumento más preciado: **la brújula.** Ya tenía todo lo necesario.

Quinto día:

Comienza la huida.

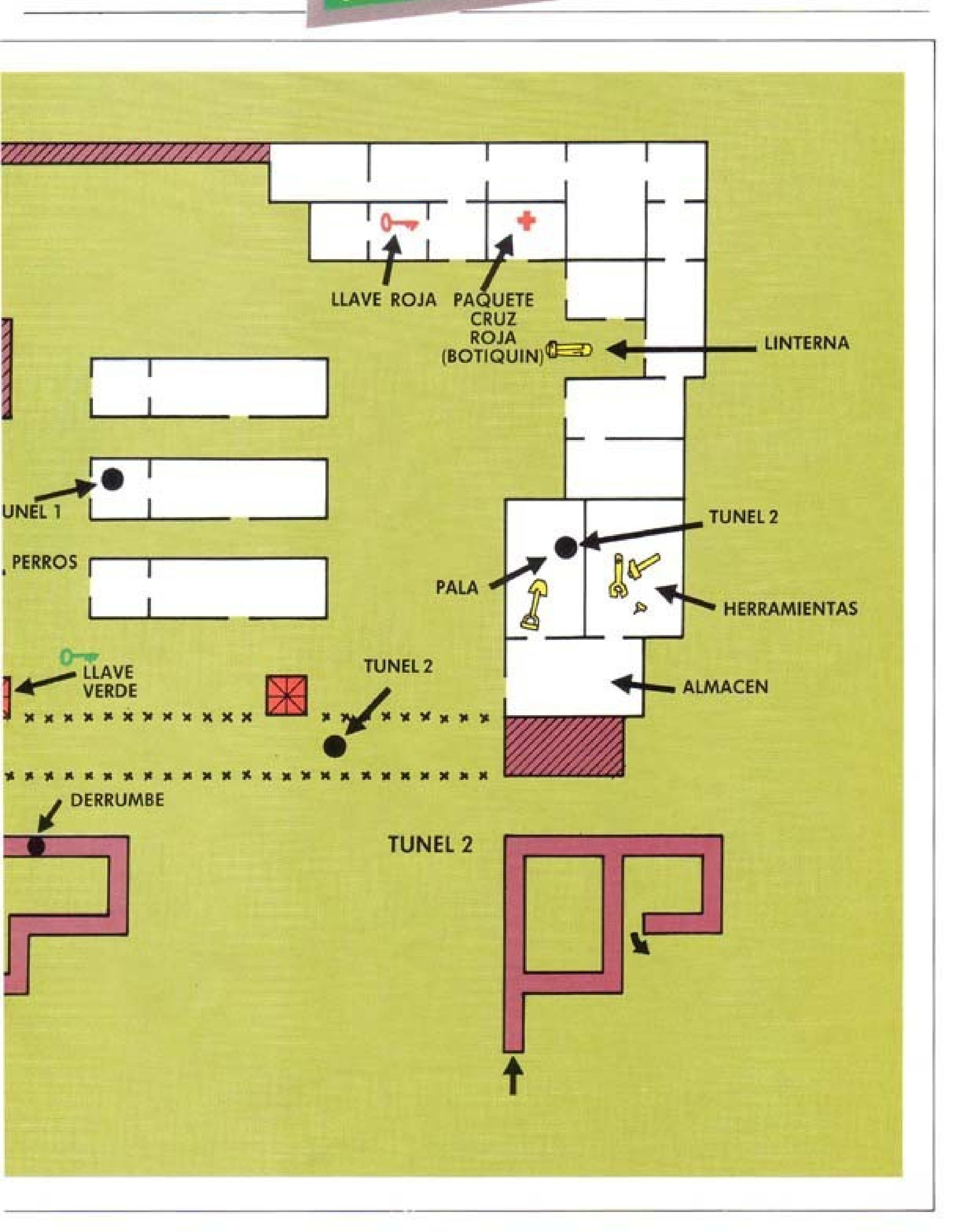
Usando la linterna para iluminar las galerías del túnel, gateó hasta el final del recorrido, donde se encontró con un inoportuno derrumbe. Después de limpiar los cascotes y despejar el paso (gracias a la eficaz ayuda de la pala), volvió a por la brújula y la bolsa, únicos objetos necesarios para



realizar la fuga con éxito. Salió al campo de ejercicios y cortó la alambrada con las tenazas. Después de dejar la brújula escondida fuera del campo, volvió a entrar y repitió la operación llevando consigo la bolsa. Una vez que logró estar afuera con los dos objetos primordiales, corrió todo lo que pudo alejándose del campo. La fuga había sido un éxito...

...Desgraciadamente, unos meses después fue capturado mientras intentaba seducir a una aldeana tirolesa, y conducido de nuevo... Suponemos que podemos ahorraros el resto.




LLAVE ROJA
PAQUETE CRUZ ROJA (BOTIQUIN)
LINTERNA
UNEL 1
PERROS
TUNEL 2
PALA
HERRAMIENTAS
LLAVE VERDE
TUNEL 2
ALMACEN
DERRUMBE
TUNEL 2

EL ZOCO

CAMBIO C-64 con cables, fuente de alimentación, DATASSETTE especial, 4 libros sobre el BASIC del Commodore, 300 juegos comerciales, Joystick QUICKSHOT II, INTERFACE COPIADOR y 15 revistas, todo en perfecto estado por un AMSTRAD con monitor incluido. Llamar al (94) 676 02 55 de Bizkaia, preguntar por Íñigo.

VENDO Spectrum Plus, cassette, interface y Joystick, más de 50 revistas, muchos juegos, libros sobre lenguajes y utilidades del Spectrum. Prefiero venderlo todo en bloque. José Ignacio, Guda Basterrechea, c/ Ntra. Sra. del Carmen n.° 15. 28039 - MADRID. Telf.: (91) 270 12 25 (llamadas de 15,00 h hasta 22,00 h).

CAMBIO juegos para Spectrum Plus. Tengo muchos y buenos. Sólo para gente de MADRID. Interesados mandar lista a: Andrés Rubio Santos, c/ Ribera del Manzanares, n.° 111, 2.° D. 28008 - MADRID. Telf.: 242 19 88.

VENDO los números del 1 al 13 incluido el extra de verano de INPUT SINCLAIR o bien cambiarlos por sus equivalentes al MSX. (Todos por 4.000 ptas.) Jorge Lucena Luque, Barrio San José obrero, blq. 43, 1.° 2.ª Telf.: 34 21 95. Reus. Tarragona.

INTERCAMBIO juegos y programas para ZX Spectrum Plus o 48K entre otros (Movie, West Bank, Dynamite Dan, Phantomas II, Spindizzy, Commando, Sky fox, Amazon, Tommy, etc.). Jaime Martínez Vanaclocha, c/ Colón, 27-2.° 46240 Carlet. Valencia.

VENDO juegos de Spectrum 15 por 1.500 ptas. cada uno. Rallie, Asterix... etc. José M.ª Olivera. Telf.: 873 36 23. Manresa. Barcelona.

INTERCAMBIO programas para el ZX Spectrum. Escribir a: J. Carlos Martín Sánchez. Restaurante Gran Sambernardo. Telf.: (988) 72 59 49 o 72 40 07. Palencia (34002).

CAMBIO C-64 con cables, fuente de alimentación, DATASSETTE especial, 4 libros sobre el BASIC del C-64, 300 juegos comerciales, Joystick QUICKSHOT II, Interface Copiador y 15 revistas, todo en perfecto estado por un AMSTRAD con monitor. Llamar al telf.: (94) 676 02 55. Bizkaia. Preguntar por Íñigo.

COMPRO el juego de COMMANDO para el Spectrum. Precio máximo 500 ptas. Manuel García, c/ Sr. de la Humildad, n.° 17. Marchena. Sevilla.

INTERCAMBIAMOS ideas, instrucciones, mapas, juegos, etc. Prometemos contestar todas las cartas. Compramos las instrucciones del Baodcopy. Abstenerse de escribir los «vendedores de programas». Club Poti Soft. Bda/Torresoto, c/ Triana, 4. Jerez de la Frontera. Cádiz. Telf.: (956) 32 12 34.

CLUB B.C.S. (se ha formado en Barcelona) abarca Spectrum, Commodore, Amstrad y MSX. Interesados escribir al apdo. de Correos de Barna, n.° 2309. C.P.: 08080 (indicando modelo de ordenador) o llamar al (93) 309 56 52. Preguntar por Ángel.

INTERCAMBIO juegos para Spectrum (novedades) (400 ptas. negociables). Gauntlet, Super Cycle, Army Moves, etc. Llamar al telf.: 47 65 66. (Sólo zona de Valladolid capital.)

VENDO Spectrum 48K, con cables, transformador, cinta Horizontes, revistas, joystick, interface y más de cincuenta juegos. Todo por 23.000 ptas. Raúl Pajarín, c/ Padre Marcellán KASAN b, 1.° F. Zaragoza. C.P.: 50015.

INTERCAMBIO programas para el ZX-Spectrum, escribir a: Juan Carlos Martín Sánchez. Restaurante Gran Sambernardo. Telf.: (988) 72 59 49 o 72 40 07. Palencia (34002).

VENDO Spectrum Plus con cassette, lápiz óptico y los juegos: Kung-Fu Master, Abu Simbel, Ghost Busters, Bounty Bob, cables y manual todo por 19.000 ptas. Telf.: 22 24 62. Tarragona.

INTERCAMBIO juegos últimas novedades 950 juegos. Ricardo Fortea, c/ Concepción Arenal, 15-1.° izq. Miranda de Ebro. Burgos.

VENDO Spectrum Plus, poco uso. Transformador, cables de conexión, cinta de demostración, manual en castellano. Incluyo un cassette grabador Sanyo (estilo periodista) pilas y corriente, incluyo transformador del cassette. Un joystick Quick Shot II con su Interface Kempston, un lote de 10 cintas de juegos con programas originales y varias revistas completamente nuevas. Todo por 40.000 ptas. Joseba Donnay, c/ Rontegui 5, 4.° dcha. Barakaldo. Vizcaya. Telf.: (94) 437 13 25 o 438 80 12 (mañanas).

VENDO Spectrum más joystick Quick Shot II, interfaces Kempston y programable, manuales en inglés y castellano y un libro sobre el Spectrum. Por 25.000 ptas. Llamar a Daniel Costa Royo. Telf.: (93) 308 25 59. Barcelona.

VENDO ordenador Spectrum ZX Plus más cassette e interface tipo Kempston todo por 27.000 ptas.; además regalo juegos. Me gustaría conseguir un ensamblador y desensamblador, tengo para cambiar programas como, Paper Boy y Ghos Goblins. Javier Vega Aguirre, c/ Cervantes, 29-2.° H. Mollet del Vallés. Barcelona - 08100. Telf.: (93) 593 65 89.

AVISO A LOS LECTORES:

El interesante programa de tratamiento de textos, cuyas dos primeras partes se publicaron en los números 14 y 15 de INPUT, se terminará de publicar, con la que será la tercera parte, en el próximo número.

La maquina alucinante







EL UNICO ORDENADOR CON MILES Y MILES DE PROGRAMAS DISPONIBLES.





Al comprar tu nuevo Spectrum pide el Pasaporte Fantástico. Podrás conseguir un reloj alucinante.

33.900 Pts.+ IVA

Microprocesador Z80A. 128 K RAM. 32 K ROM. Teclado de 58 teclas. 32 columnas × 24 filas de texto. Gráficos de alta resolución (256 × 192 pixels). 8 colores con dos niveles de brillo cada uno. Calculadora en pantalla. 3 canales de sonido programables e independientes. Cassette incorporada. Salida TV y monitor RGB.

Interface MIDI (Musical Instrument Digital Interface). Salida Serie RS 232 bidireccional. Dos conectores para joysticks. Conector plano compatible con todos los modelos Spectrum anteriores. Editor de pantalla y dos versiones BASIC en ROM. 48 K BASIC, compatible con Spectrum 16 K, 48 K y ZX + . 128 K BASIC, compatible con ZX Spectrum 128.

Nuevo sinclair ZX Spectrum +2

C/. Aravaca, 22. 28040 Madrid. Tel. 459 30 01. Telex 47660 INSC E. Fax 459 22 92. Delegación en Cataluña: C/. Tarragona, 110. Tel. 325 10 58. 08015 Barcelona.